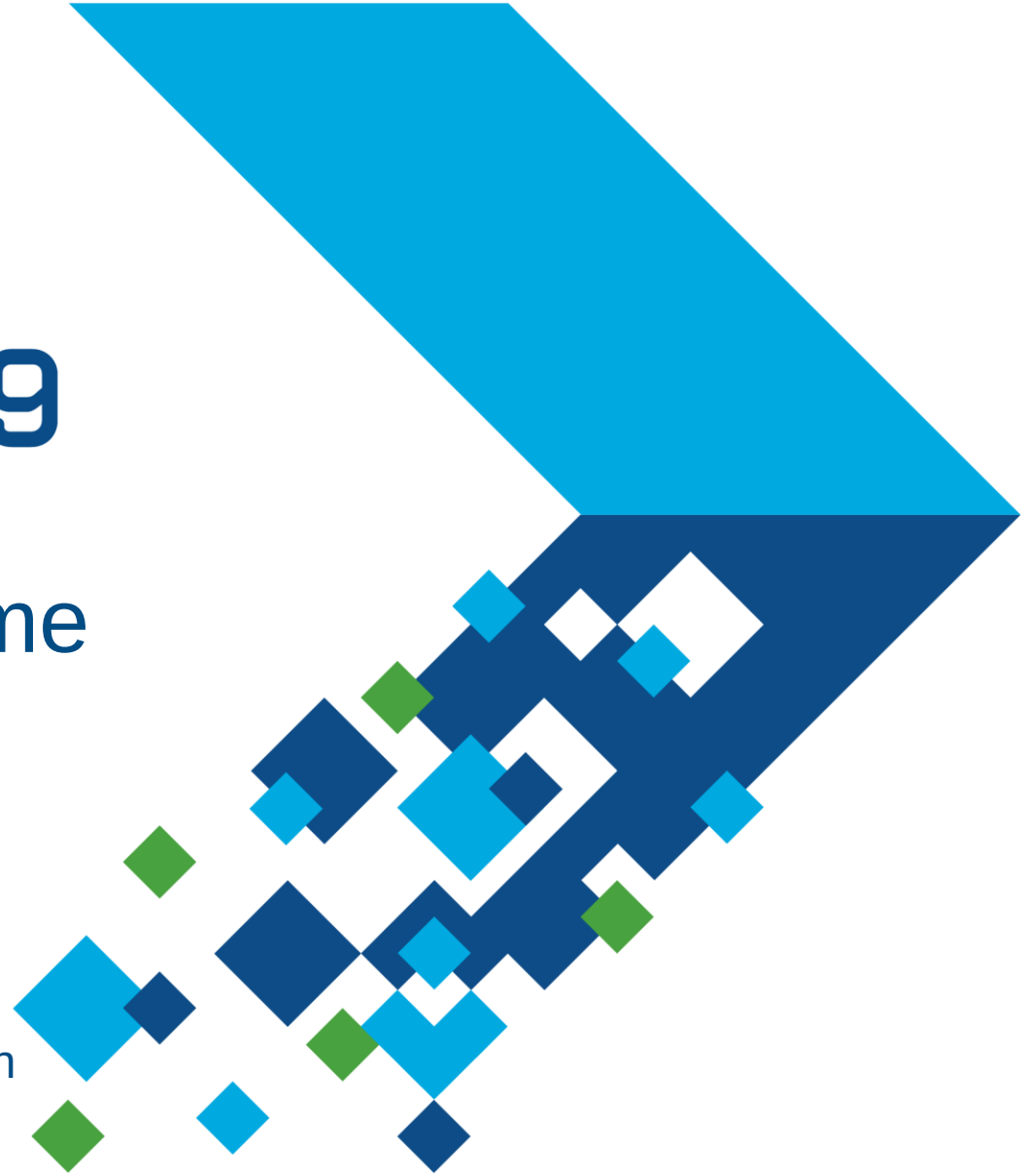
MATLAB EXPO 2019

Deploying AI for Near Real-Time Manufacturing Decisions

(Masterclass)

Pallavi Kar
Application Engineer – Data Science & Enterprise Integration



Digital Transformation and IIoT

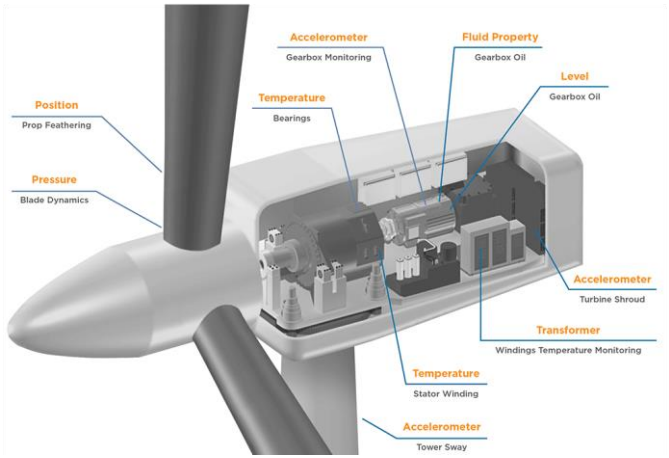
Customer Goal

By **connecting machines** in operation,

you can use **data, algorithms, and models**

to make **better decisions**, improve processes, reduce cost, improve customer experience.

- **Industrial IoT**
- **Digital Twin**
- **Industry 4.0**
- **Smart ‘XYZ’**
- **Digital Transformation**



Predictive Maintenance

- Operating Conditions vary over time and location
- Component Life and Safety

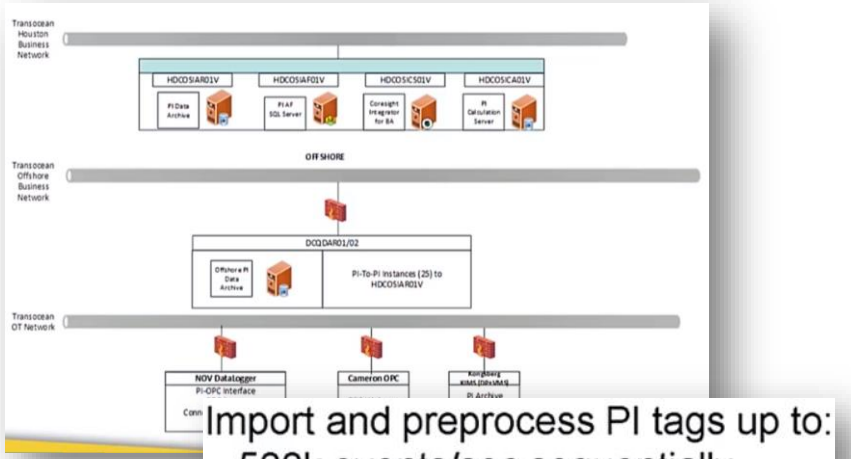
Transocean performed CPM of a BOP using an adaptive physics-based modeling approach with Simscape Link

Case Study: Transocean

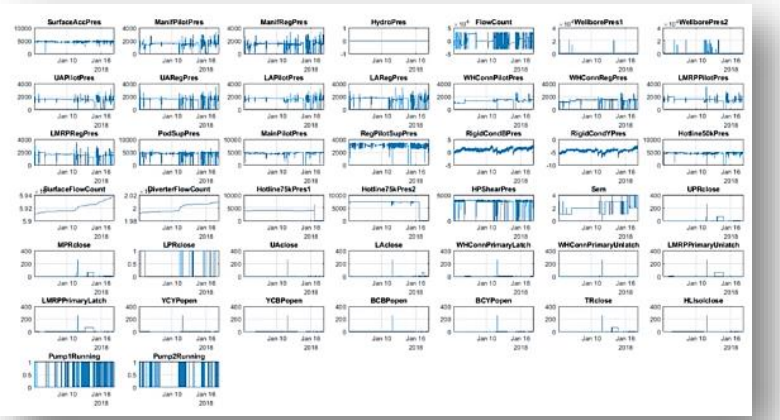
Objective: Reduce BOP downtime

Solution:
- Simulink model of BOP and Control System
- Simulate 100s future scenarios - degradation trends and anomalies
- Pi Servers to collect data
- Preprocess data to avoid noise and outliers
- Train Models on future scenarios to predict in advance

Outcome: Robust condition and performance monitoring of BOP reduced downtime



Import and preprocess PI tags up to:
500k events/sec sequentially
1,500k events/sec in parallel



Monitor → Analyze → Predict → Control → Optimize

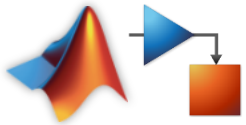
Create Digital Twin → Use Digital Twin

Example Problem: Develop and operationalize a digital twin and a machine learning model to predict failures in industrial pumps

Current system requires Operator to manually monitor operational metrics for anomalies. Their expertise is required to detect and take preventative action

Process Engineer

Develops models in MATLAB and Simulink



System Architect

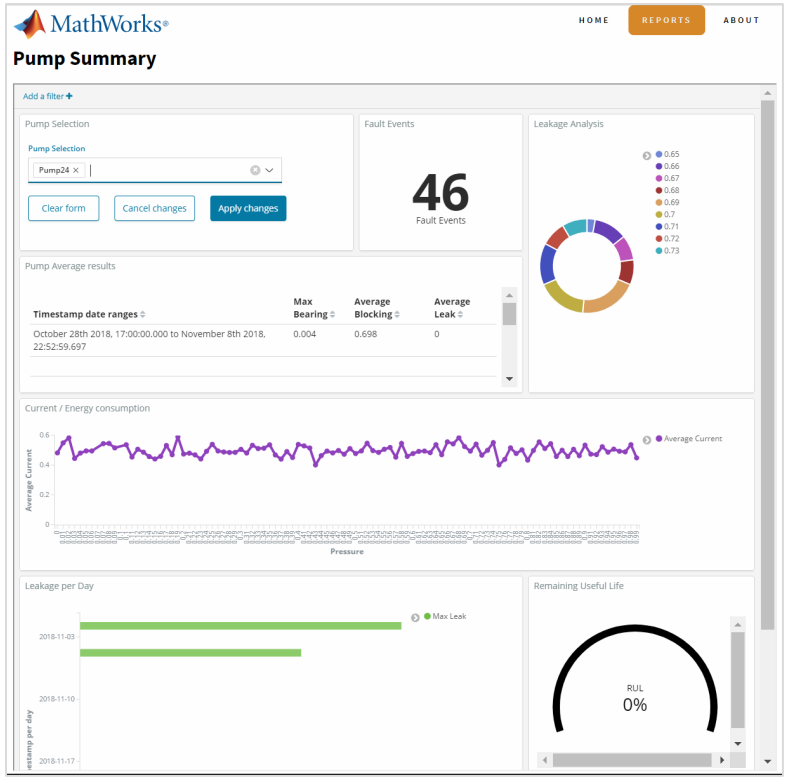
Deploys and operationalizes model on Azure cloud

Azure

Operator

Makes operational decisions based on model output

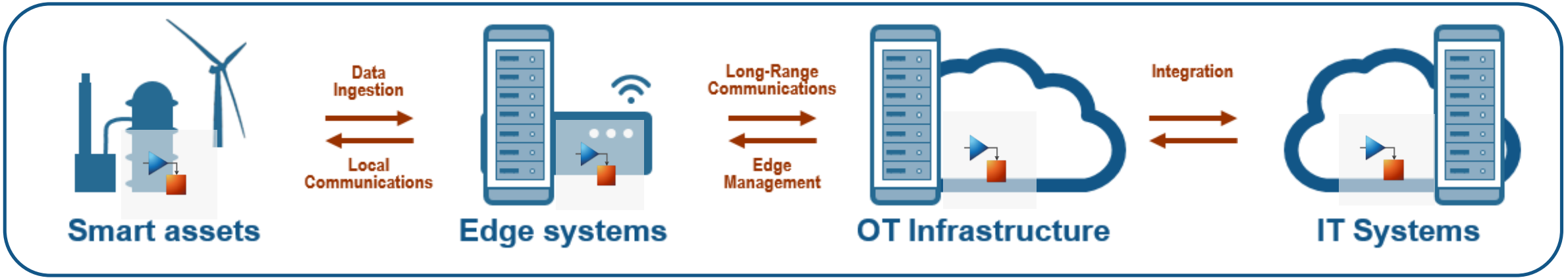
kibana

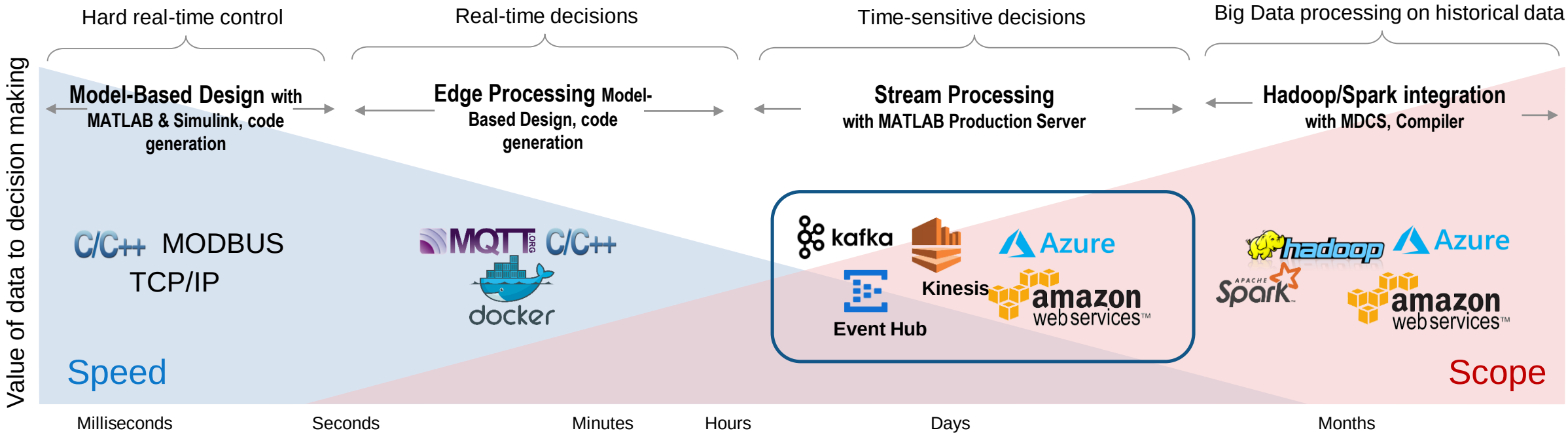


Predictive Maintenance Workflow

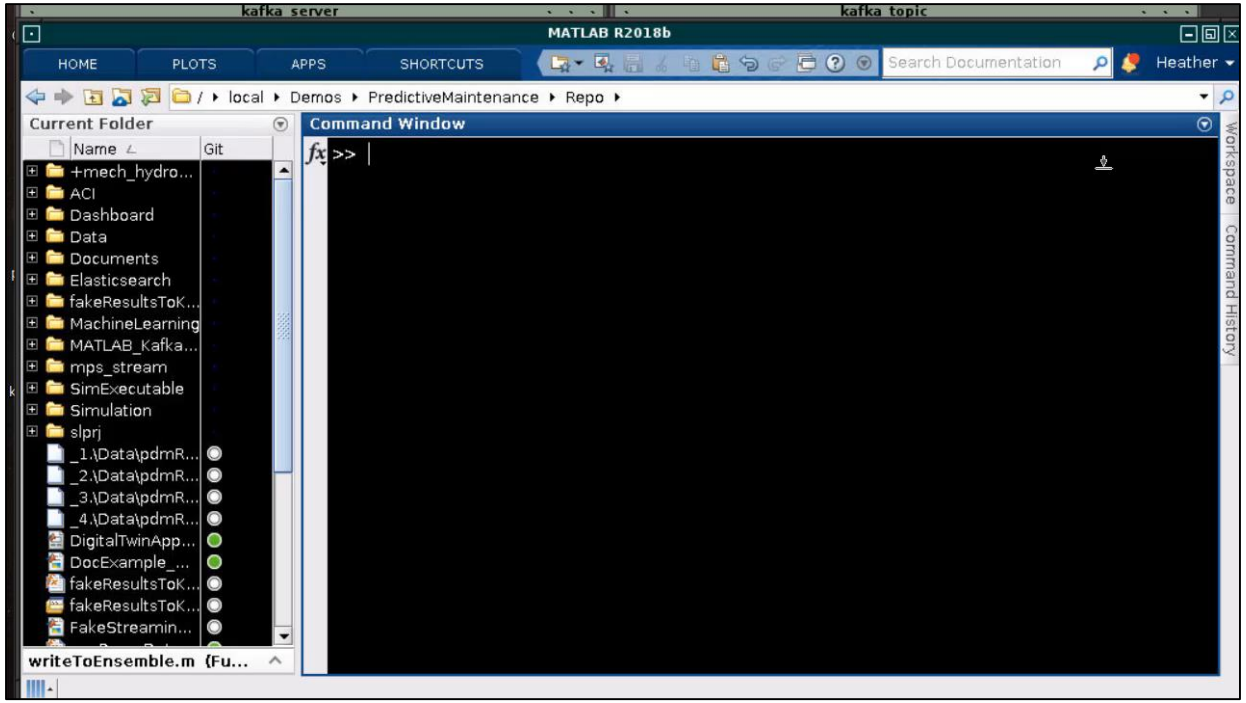
1. **Access and Explore Data**
   - Files
   - Databases
   - Sensors
2. **Preprocess Data**
   - Working with Messy Data
   - Data Reduction/ Transformation
   - Feature Extraction
3. **Develop Predictive Models**
   - Model Creation e.g. Machine Learning
   - Parameter Optimization
   - Model Validation
4. **Integrate with Production Systems**
   - Desktop Apps
   - Enterprise Scale Systems (kafka, AWS Kinesis, OSIsoft)
   - Embedded Devices and Hardware
5. **Visualize Results**
   - 3rd party dashboards (Spotfire TIBCO Software, tableau, Qlik, Power BI)
   - Web apps

Backbone Infrastructure for *Preventive, Predictive, Reactive, Actionable Analytics*

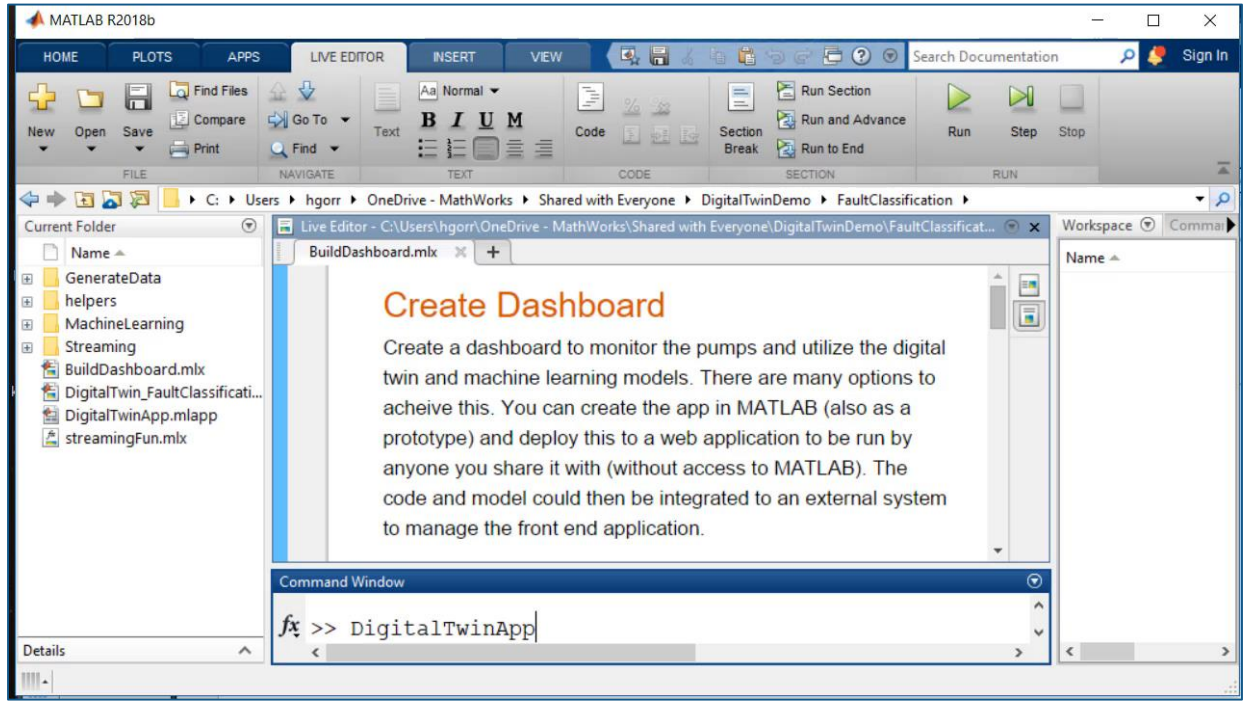




Steaming Analytics - Remaining Useful Life



Edge Device Publishing Data



Consume data and Update RUL

Project statement: Develop end-to-end predictive maintenance system

Plant Operator

1. Monitor ***flow***, ***pressure***, and ***current*** of each pump so I always know their ***operational state***
2. Need an ***alert*** when fault parameters drift outside an acceptable range so I can take ***immediate corrective action***
3. Continuous estimate of each pump's ***remaining useful life (RUL)*** so that I can ***schedule maintenance or replace*** the asset

Project constraints and solutions

Process Engineer

We don’t have a large set of failure data, and it’s too costly to generate real failures in our plant for this project

Solution: Use an accurate physics-based software model for the pump to develop synthetic training sets

Project constraints and solutions

Process Engineer

Need software for multidisciplinary problem across teams, plus integration w/ IT

Solution: Use MATLAB and integrate with OSS

Project constraints and solutions

System Architect

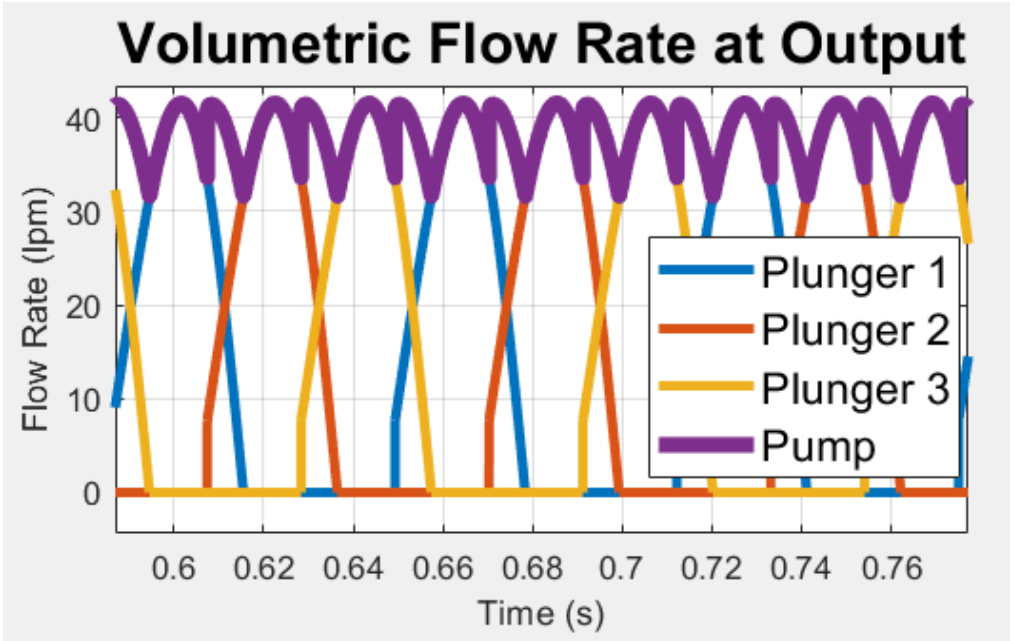
We don’t have a large IT/hardware budget, and we need to see results before committing to a particular platform or technology

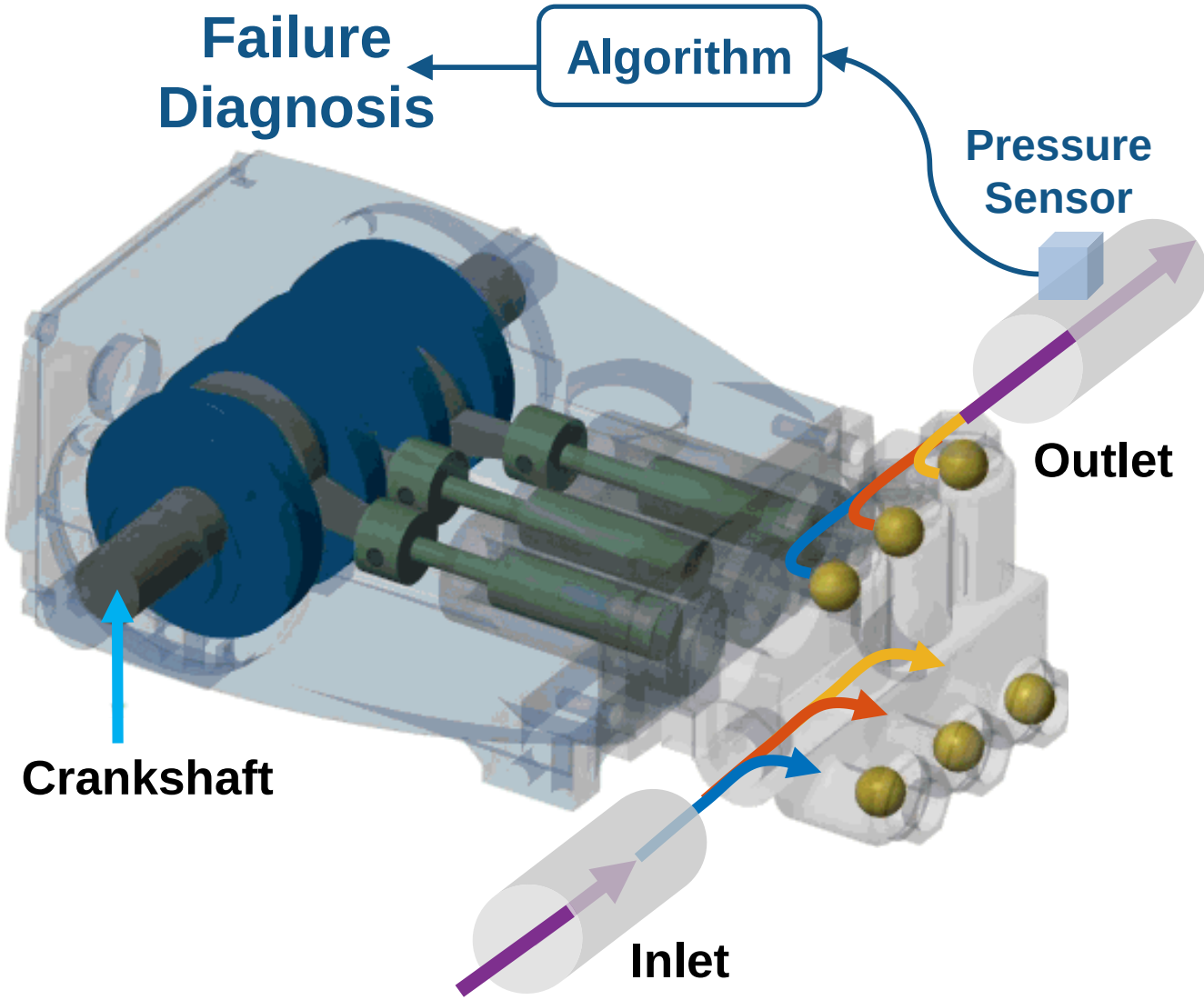
Solution: Leverage cloud platform to quickly configure and provision the services needed to build the solution, while minimizing lock-in to a particular provider

Physics of Triplex Pump

Process Engineer

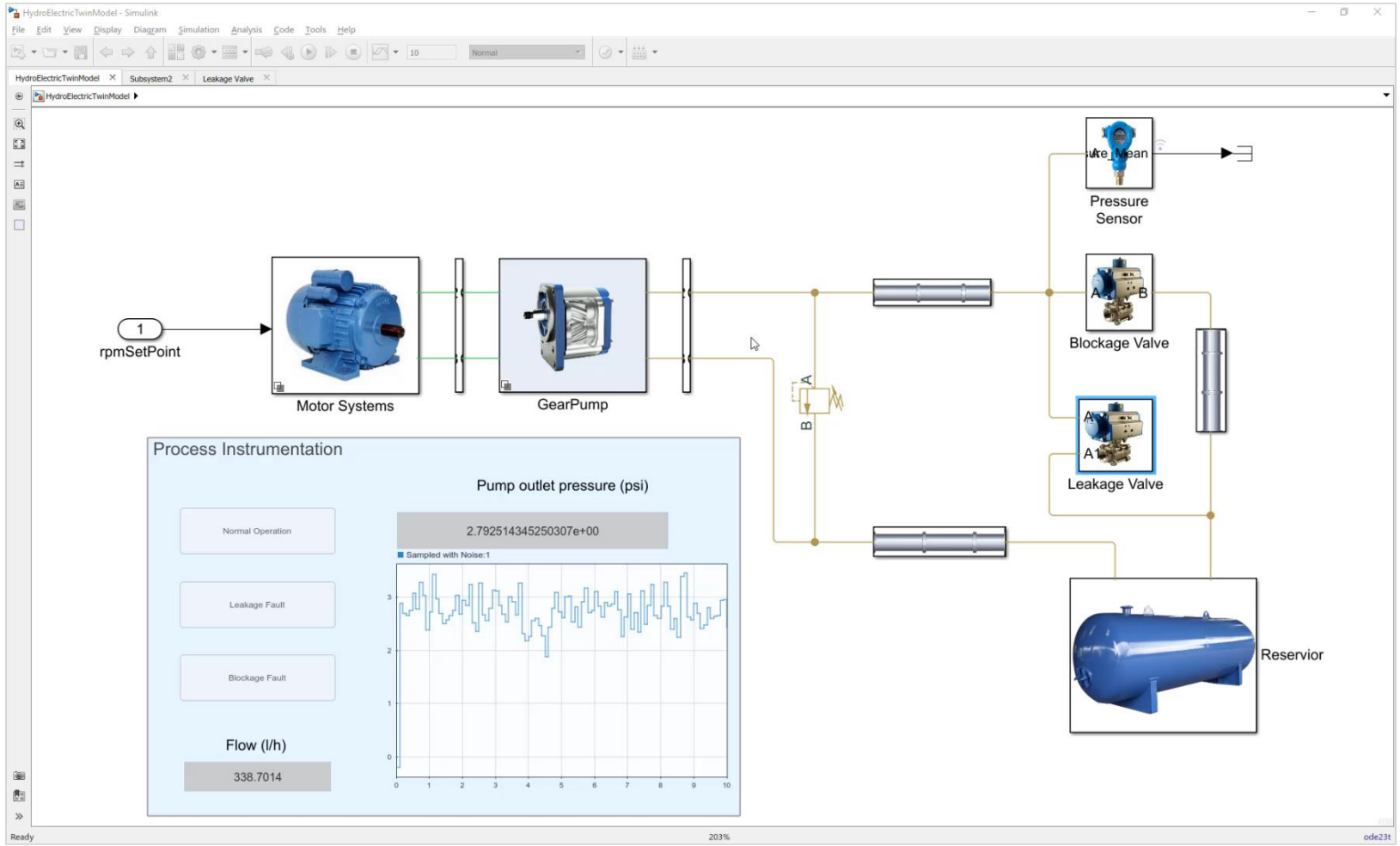
- Crankshaft drives three plungers
  - Each 120 degrees out of phase
  - One chamber always discharging
  - Three types of failures





Creating Multi-Domain Physical Models using Simscape

Monitor | Analyze | Predict | Control | Optimize



Pump Hardware

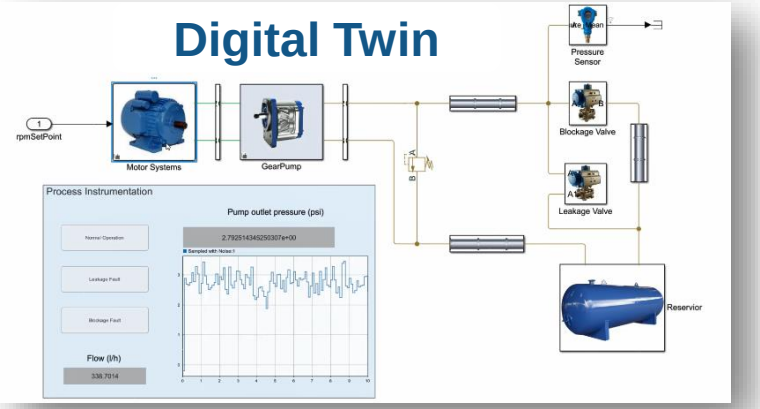
Acquire Real-Time Data for Updating Digital Twin

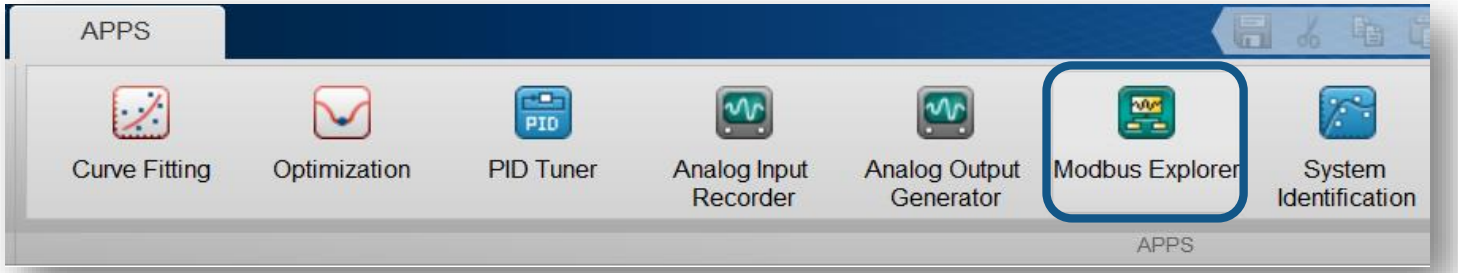
Monitor | Analyze | Predict | Control | Optimize

Pump Hardware

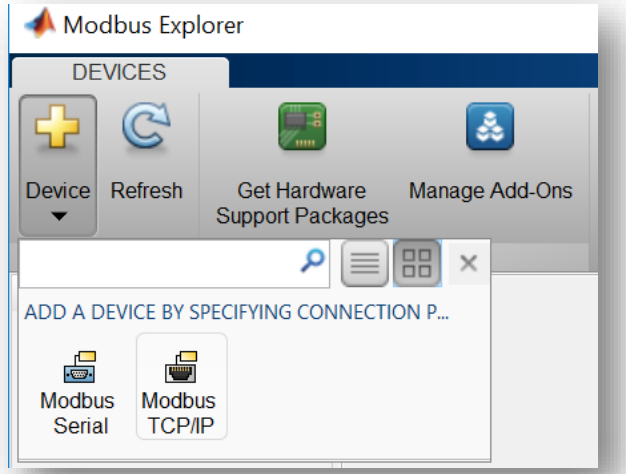
MODBUS TCPIP

Digital Twin



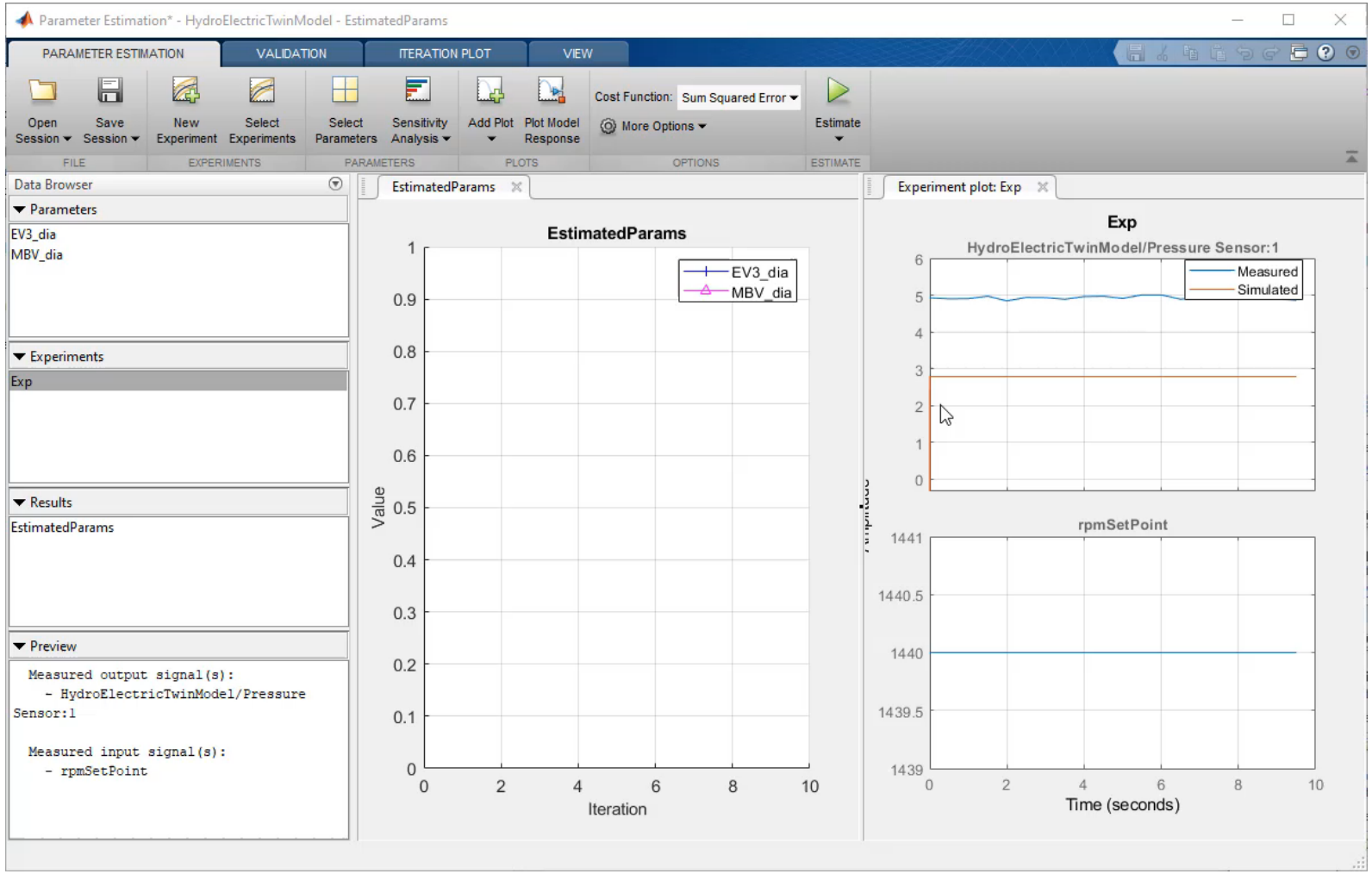


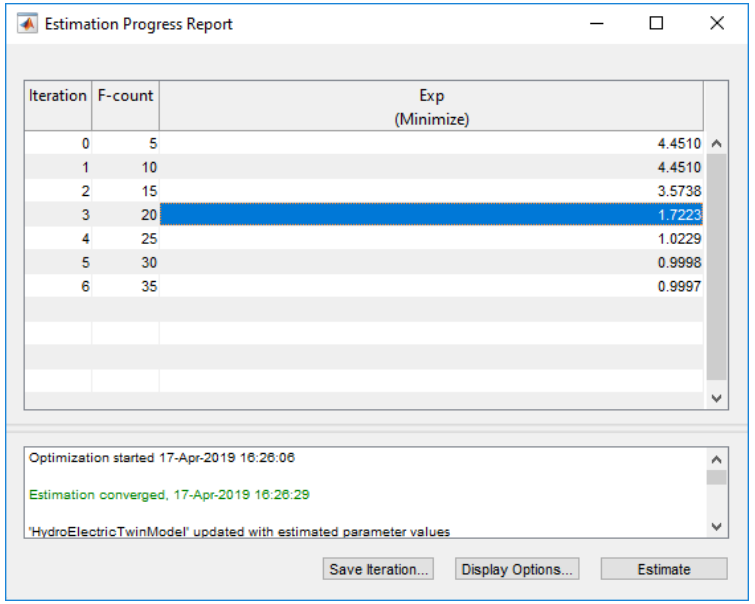

```
m = modbus('tcpip', '192.168.2.1', 308)

m =

   Modbus TCPIP with properties:

    DeviceAddress: '192.168.2.1'
             Port: 308
           Status: 'open'
       NumRetries: 1
          Timeout: 10 (seconds)
        ByteOrder: 'big-endian'
        WordOrder: 'big-endian'
```



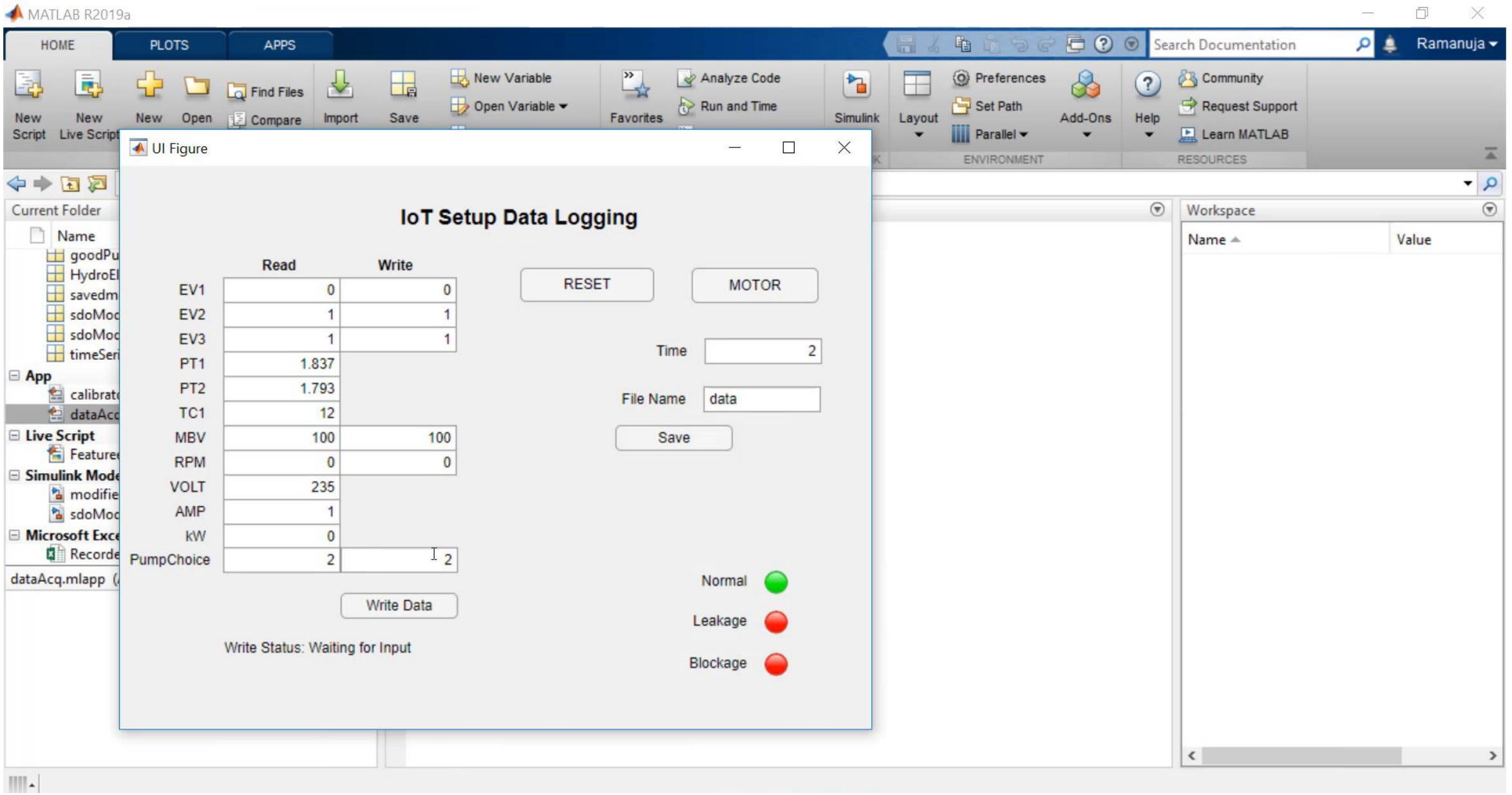
Use Simulink Design Optimizer to

Monitor | Analyze | Predict | Control | Optimize



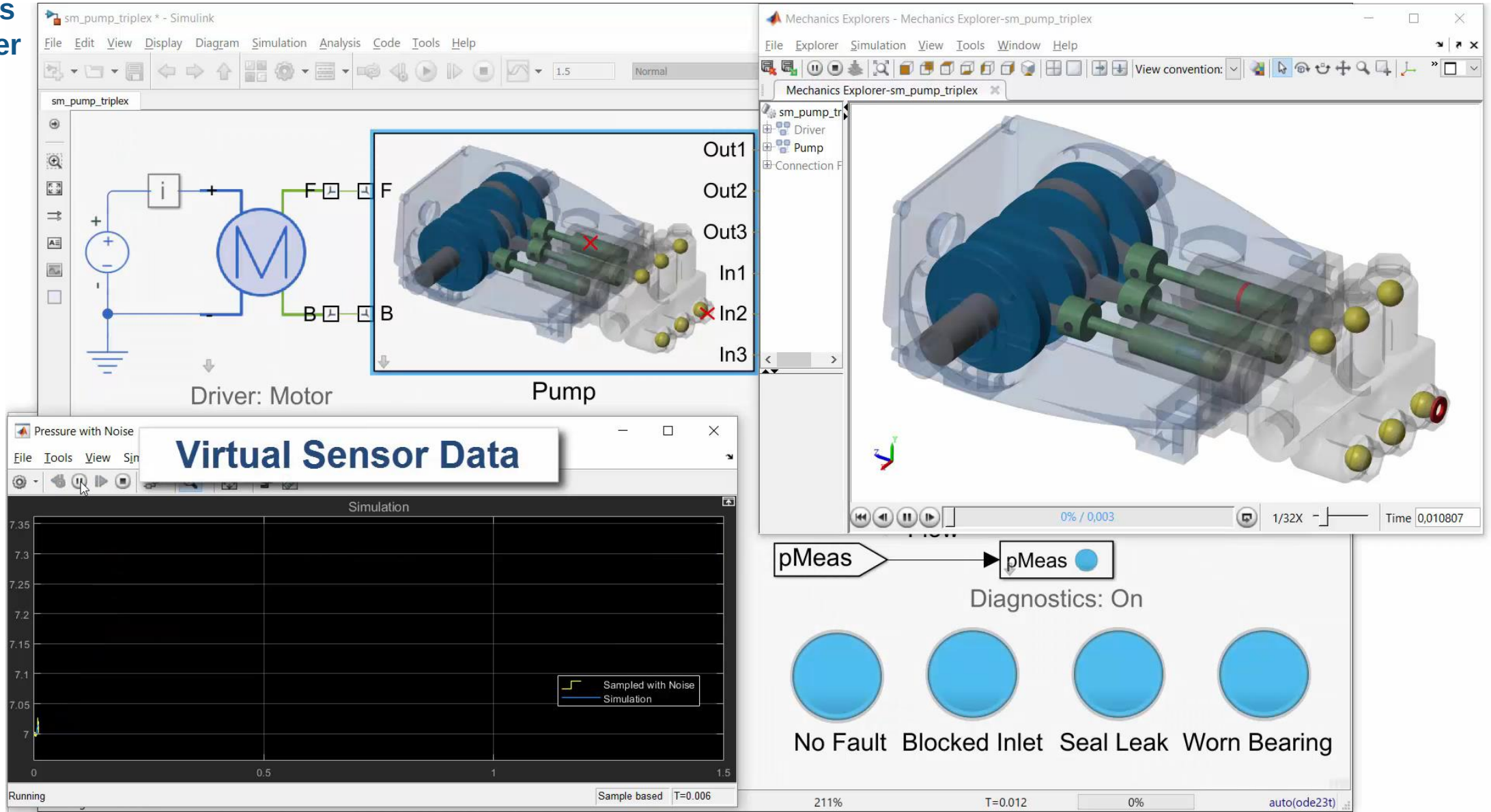


- ✓ Setup Experiments
- ✓ Parameterize
- ✓ Save Sessions
- ✓ Generate Code


MATLAB R2019a
HOME
PLOTS
APPS
Search Documentation
Ramanuja
New Script
New Live Script
New
Open
Find Files
Compare
Import
Save
New Variable
Open Variable
Favorites
Analyze Code
Run and Time
Simulink
Layout
Preferences
Set Path
Parallel
Add-Ons
Help
Community
Request Support
Learn MATLAB
ENVIRONMENT
RESOURCES
Current Folder
Name
App
Live Script
Simulink Mode
Microsoft Exce
dataAcq.mlapp
Workspace
Name
Value
UI Figure
IoT Setup Data Logging
Read
Write
EV1
EV2
EV3
PT1
PT2
TC1
MBV
RPM
VOLT
AMP
kW
PumpChoice
RESET
MOTOR
Time
File Name
data
Save
Write Data
Write Status: Waiting for Input
Normal
Leakage
Blockage

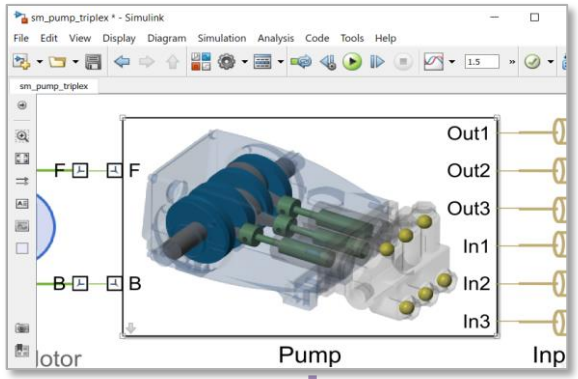
Build digital twin and generate sensor data

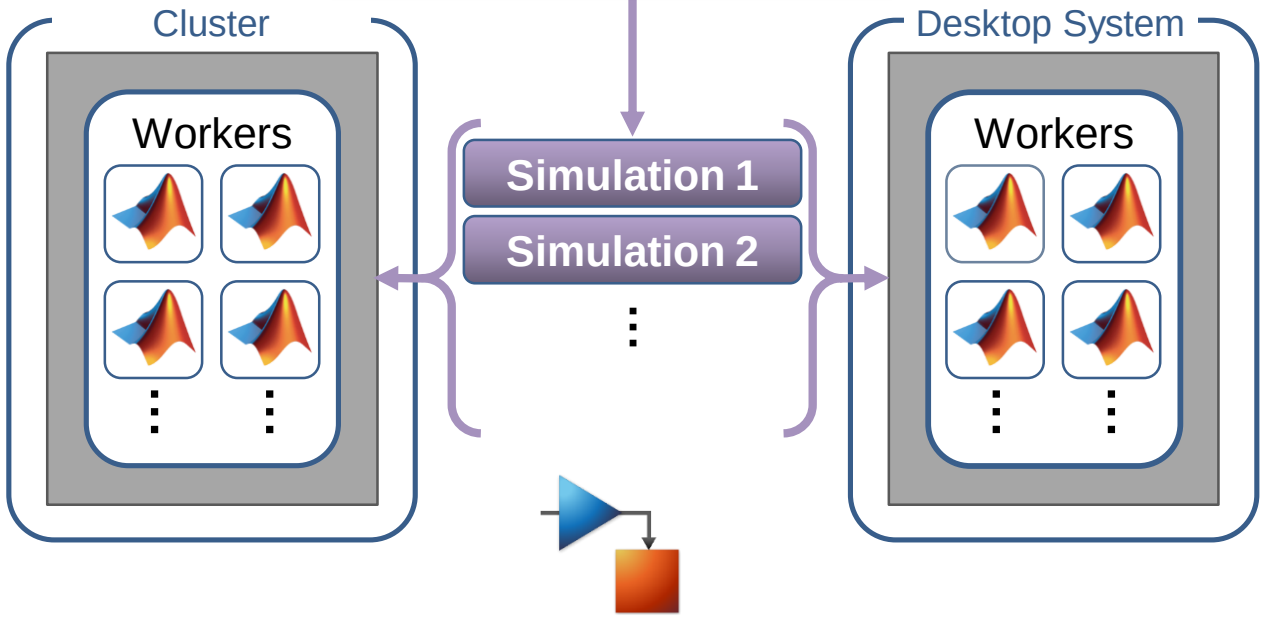
Process Engineer



Simulate data with many failure conditions

Process Engineer



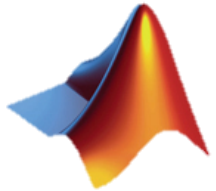


Run parallel simulations

Access Data

```
ens = simulationEnsembleDatastore(location)
```

```
ens =
  simulationEnsembleDatastore with properties:

          DataVariables: [25×1 string]
    IndependentVariables: [0×0 string]
      ConditionVariables: [0×0 string]
       SelectedVariables: [25×1 string]
                ReadSize: 1
              NumMembers: 702
          LastMemberRead: [0×0 string]
                   Files: [702×1 string]
```



Store data on HDFS

Predictive Maintenance Workflow

1 **Access and Explore Data**

- Files
- Databases
- Sensors

2 **Preprocess Data**

- Working with Messy Data
- Data Reduction/ Transformation
- Feature Extraction

3 **Develop Predictive Models**

- Model Creation e.g. Machine Learning
- Parameter Optimization
- Model Validation

4 **Integrate with Production Systems**

- Desktop Apps
- Enterprise Scale Systems: kafka, AWS Kinesis, OSIsoft
- Embedded Devices and Hardware

5 **Visualize Results**

- 3rd party dashboards: Spotfire TIBCO Software, tableau, Qlik, Power BI
- Web apps

Process Engineer

2 Preprocess Data

Represent signal information

Signal processing

```
[Spectrum,Frequencies] = pspectrum(data.Flow);
[pLow,pHigh] = bounds(Spectrum);
fPeak = Frequencies(Spectrum==pHigh);
qPeak2Peak = peak2peak(data.Flow);
qCrest = peak2rms(data.Flow);
qRMS = rms(data.Flow);
qMAD = mad(data.Flow);
```

Diagnostic Feature Designer - Signal Trace: flow/Data

FEATURE DESIGNER | SIGNAL TRACE | VIEW

RL Func | RL Examples | CleanUp

Open Session | Save Session | Import Data | Signal Trace | Computation Options | Filtering & Averaging | Residue Generation | Spectral Estimation | Time-Domain Features | Spectral Features | Rank Features | Export

FILE | PLOT | COMPUTATION | DATA PROCESSING | FEATURE GENERATION | RANKING | EXPORT

Data Browser

Signals & Spectra

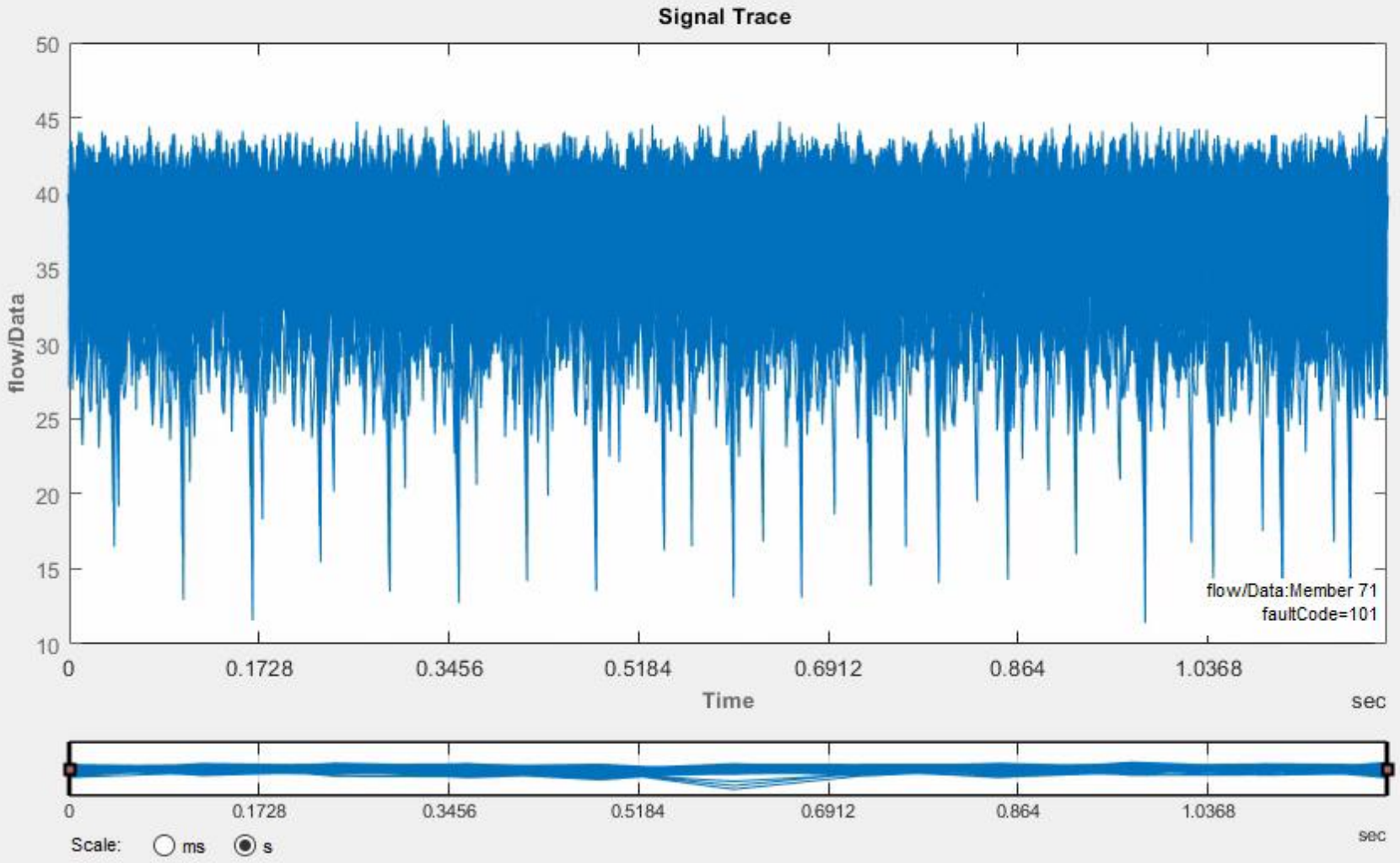
- flow/Data
- pressure/Data

Feature Tables

Datasets

- Ensemble1

Signal Trace: flow/Data



Signal trace plot for "flow/Data" is in focus.

Diagnostic Feature Designer - Signal Trace: flow/Data

FEATURE DESIGNER | SIGNAL TRACE | VIEW

RL Func | RL Examples | CleanUp

Open Session | Save Session | Import Data | Signal Trace | Power Spectrum | Order Spectrum | Histogram | Select data to plot | Computation Options | Filtering & Averaging | Residue Generation | Spectral Estimation | Time-Domain Features | Spectral Features | Rank Features | Export

FILE | PLOT | COMPUTATION | DATA PROCESSING | FEATURE GENERATION | RANKING | EXPORT

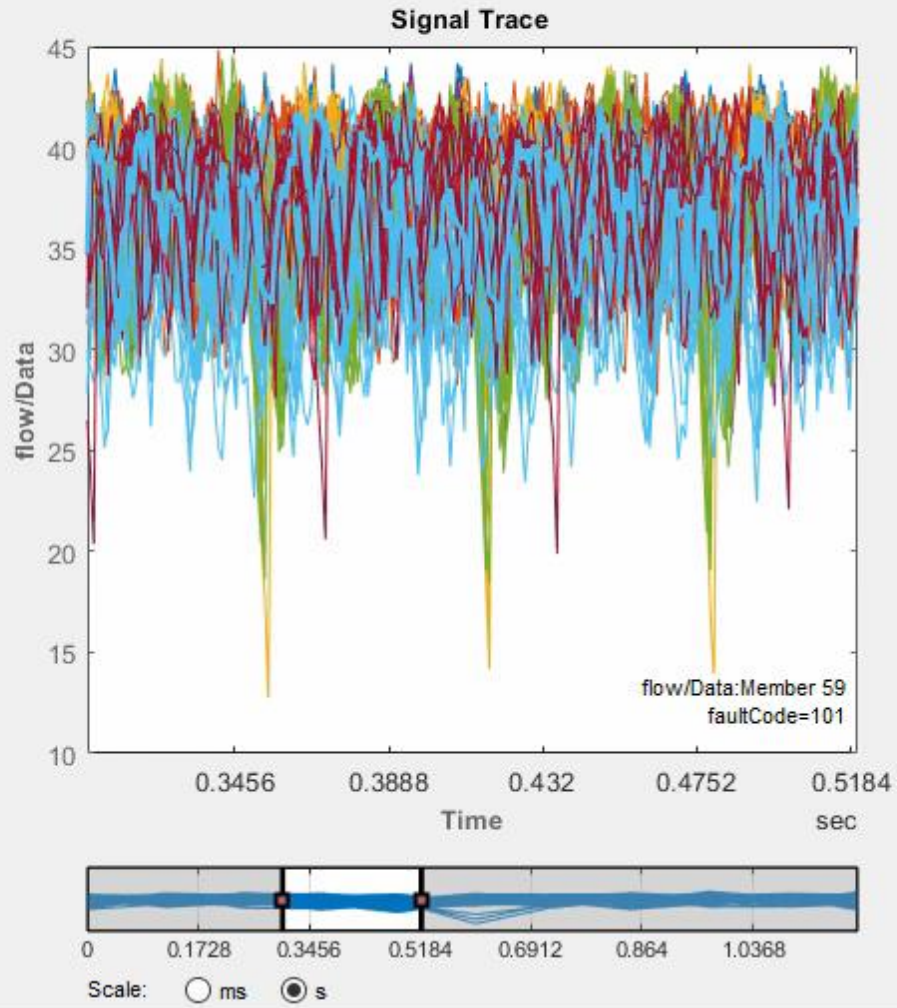
Data Browser

Signals & Spectra
- flow/Data
- pressure/Data
- flow_ps/Data

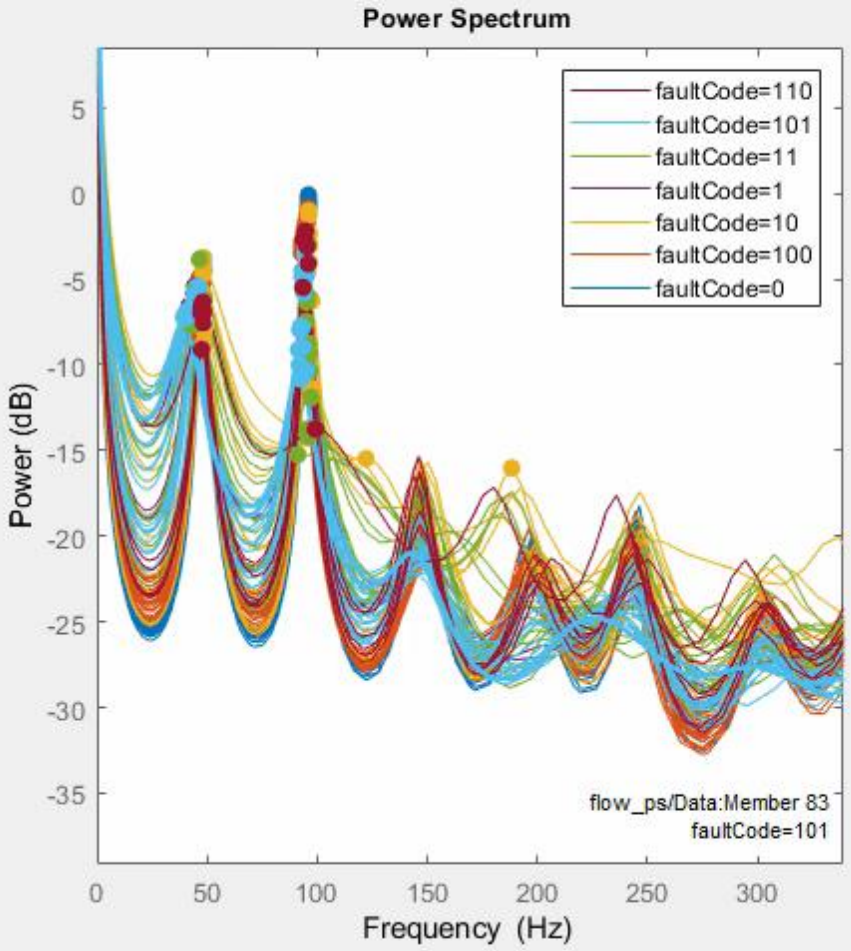
Feature Tables

Datasets
- Ensemble1

Signal Trace: flow/Data



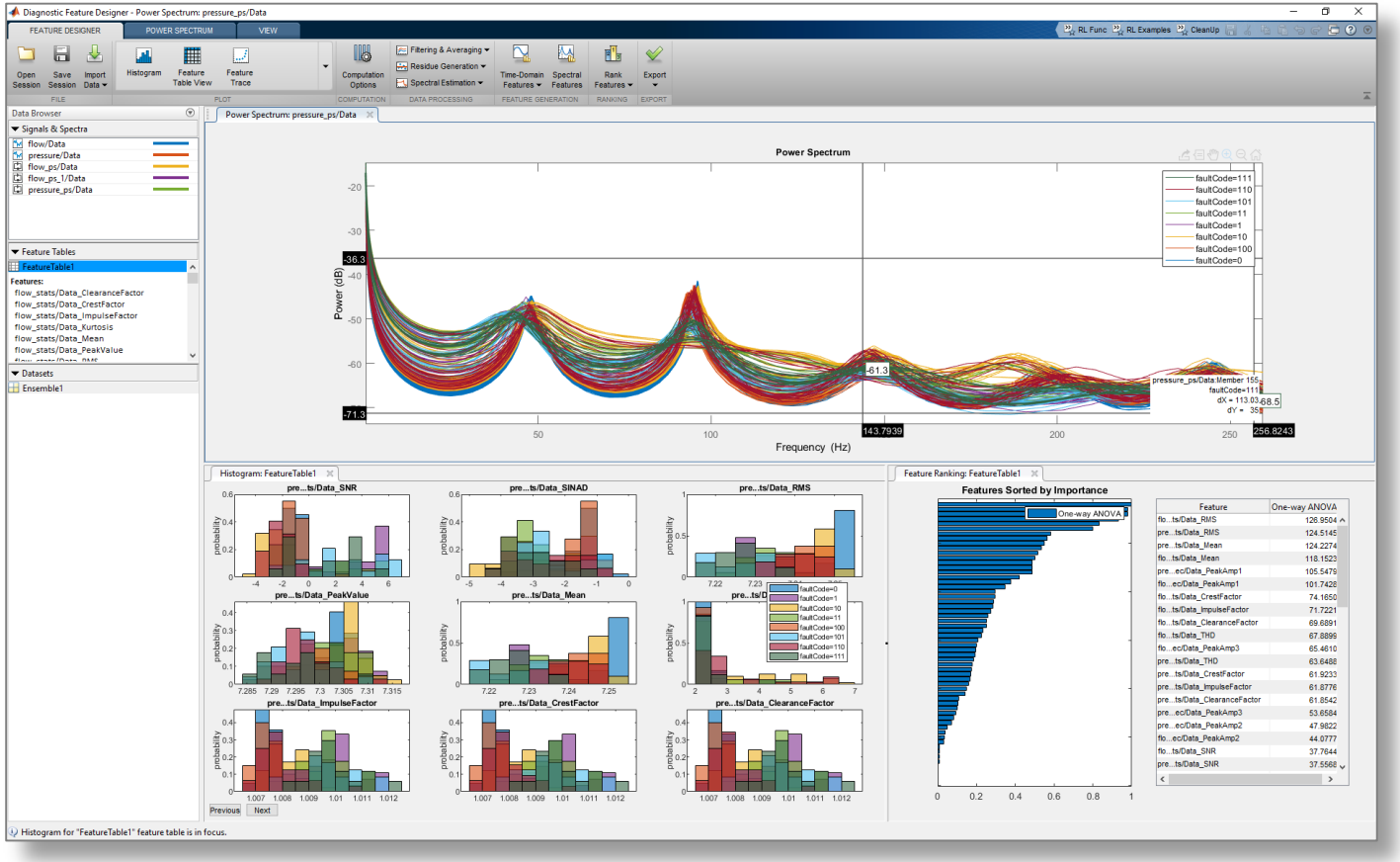
Power Spectrum: flow_ps/Data



Diagnostic Feature Designer App

Predictive Maintenance Toolbox **R2019a**

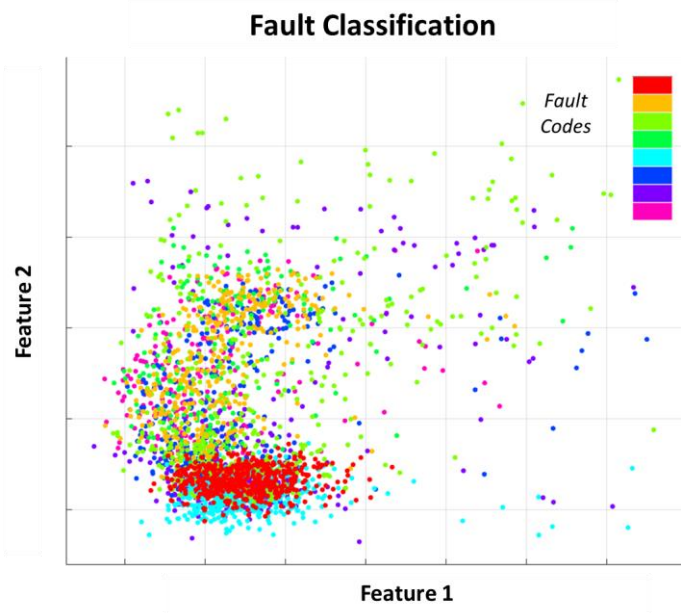
- Extract, visualize, and rank features from sensor data
- Use both statistical and dynamic modeling methods
- Work with out-of-memory data
- Explore and discover techniques without writing MATLAB code

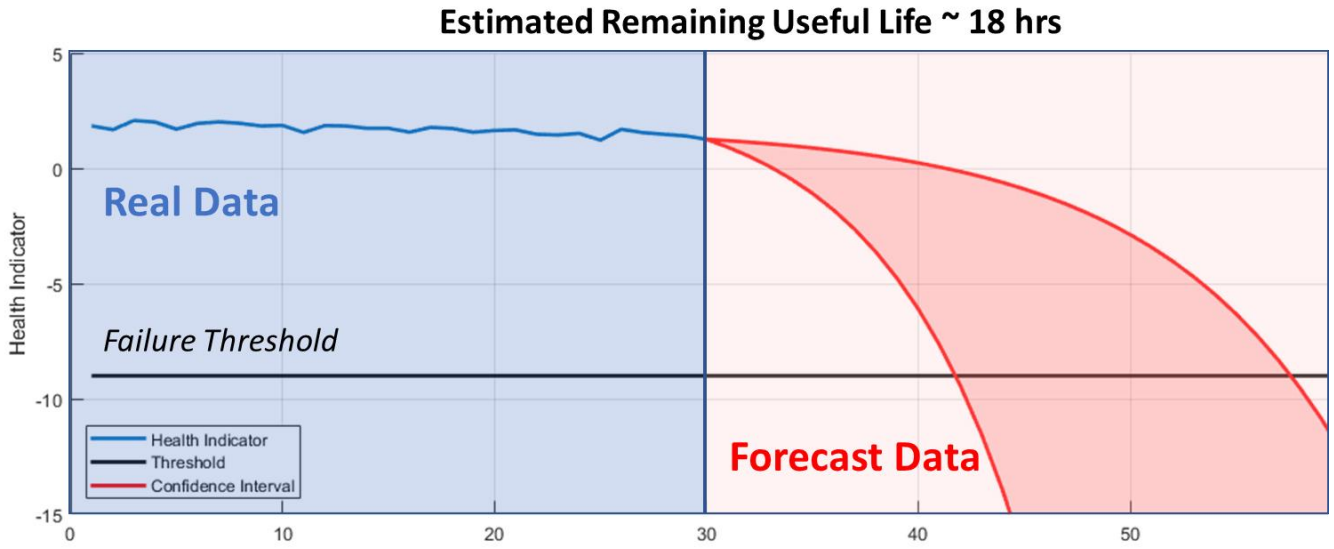


3 Develop Predictive Models

Process Engineer

Develop Predictive Models in MATLAB





Type of Fault (Classification)

Plant Operator

Remaining Useful Life (Regression)

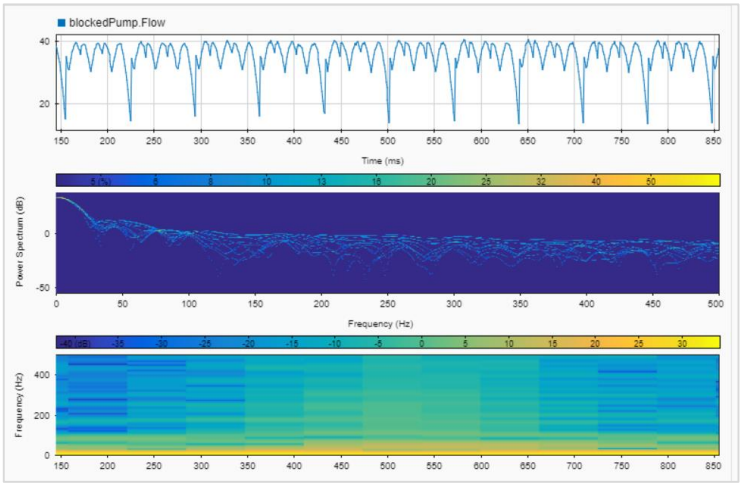
3 Develop Predictive Models

Develop Predictive Models in MATLAB

Process Engineer

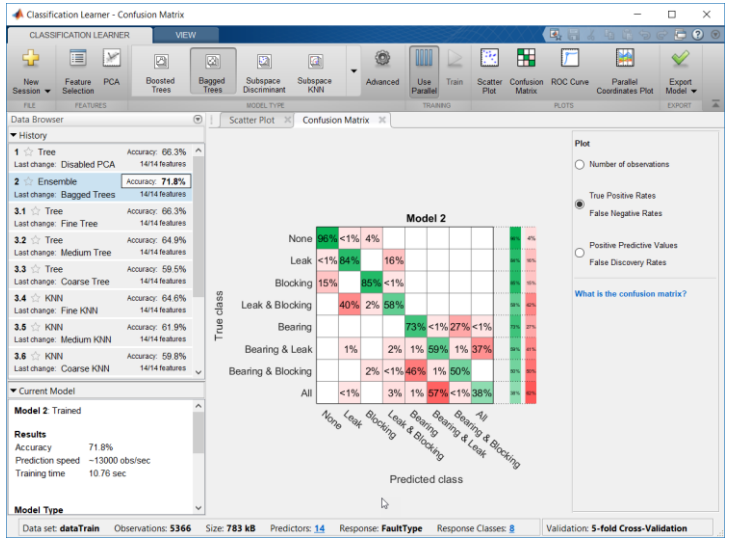
	Time	1 LeakFault	2 BlockingFault	3 BearingFault	4 FaultType
1	0 sec	2.8472	-0.1477	1.8000	All
2	0.001 sec	-0.1498	-0.4207	1.3103	Bearing & Blocking
3	0.002 sec	0.6511	1.6521	-0.5557	Leak
4	0.003 sec	0.1469	-0.2775	1.0074	All
5	0.004 sec	-0.6480	0.7065	-0.8878	Blocking
6	0.005 sec	-0.8165	-0.5434	-0.3079	Blocking
7	0.006 sec	-1.0061	1.2083	0.0661	Bearing
8	0.007 sec	1.0125	-1.9098	-0.7027	Leak & Blocking

Label Faults



Represent Signals

Train Model



Validate Model

Scale

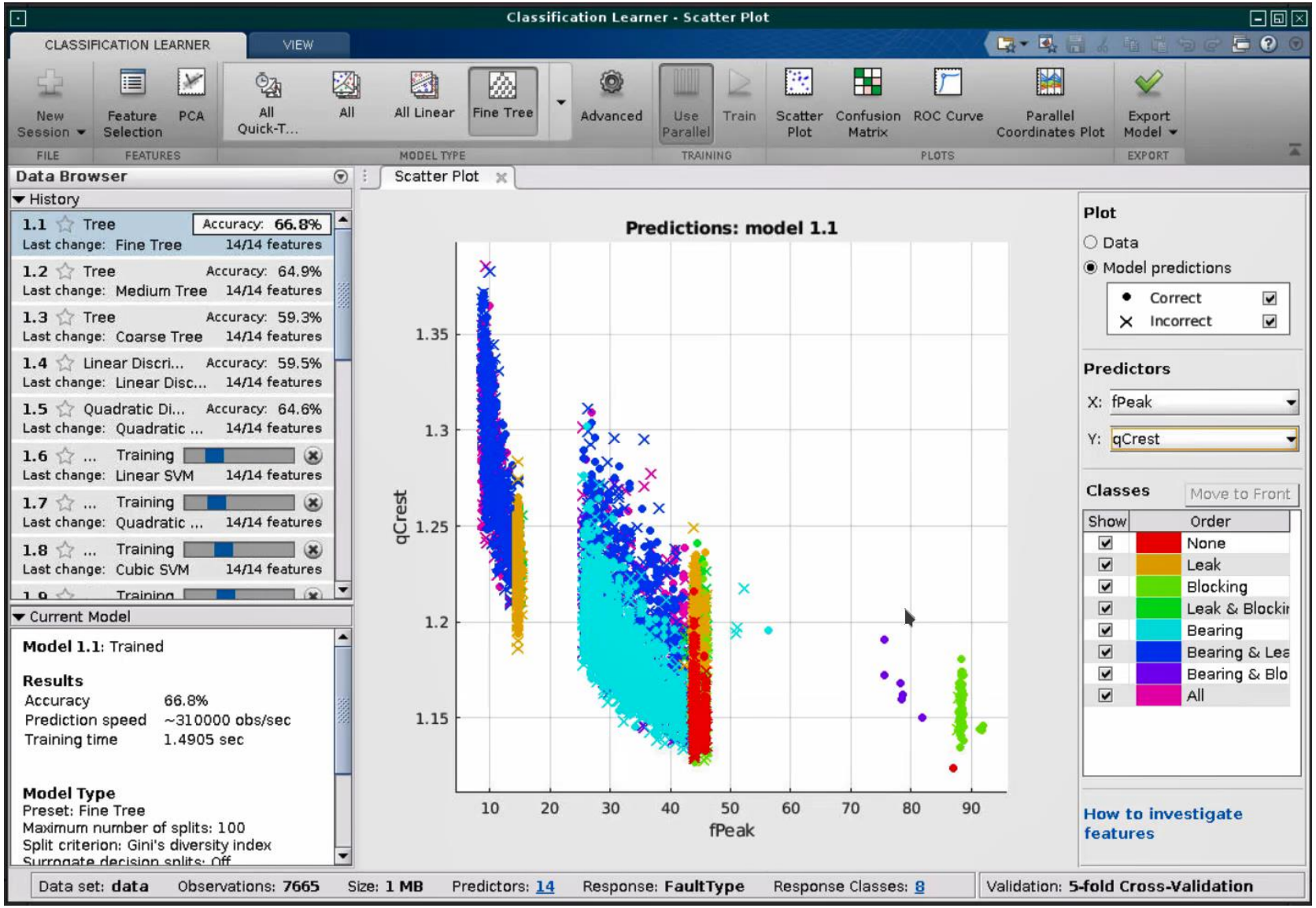
```
tt = tall(ds);
tt = preprocessData(tt);
model = TreeBagger(50,tt,'Event');
```

```
Evaluating tall expression using the Spark Cluster:
- Pass 1 of 2: Completed in 11 sec
- Pass 2 of 2: Completed in 2.3333 min
Evaluation completed in 2.6167 min
```

Process Engineer

3 Develop Predictive Models

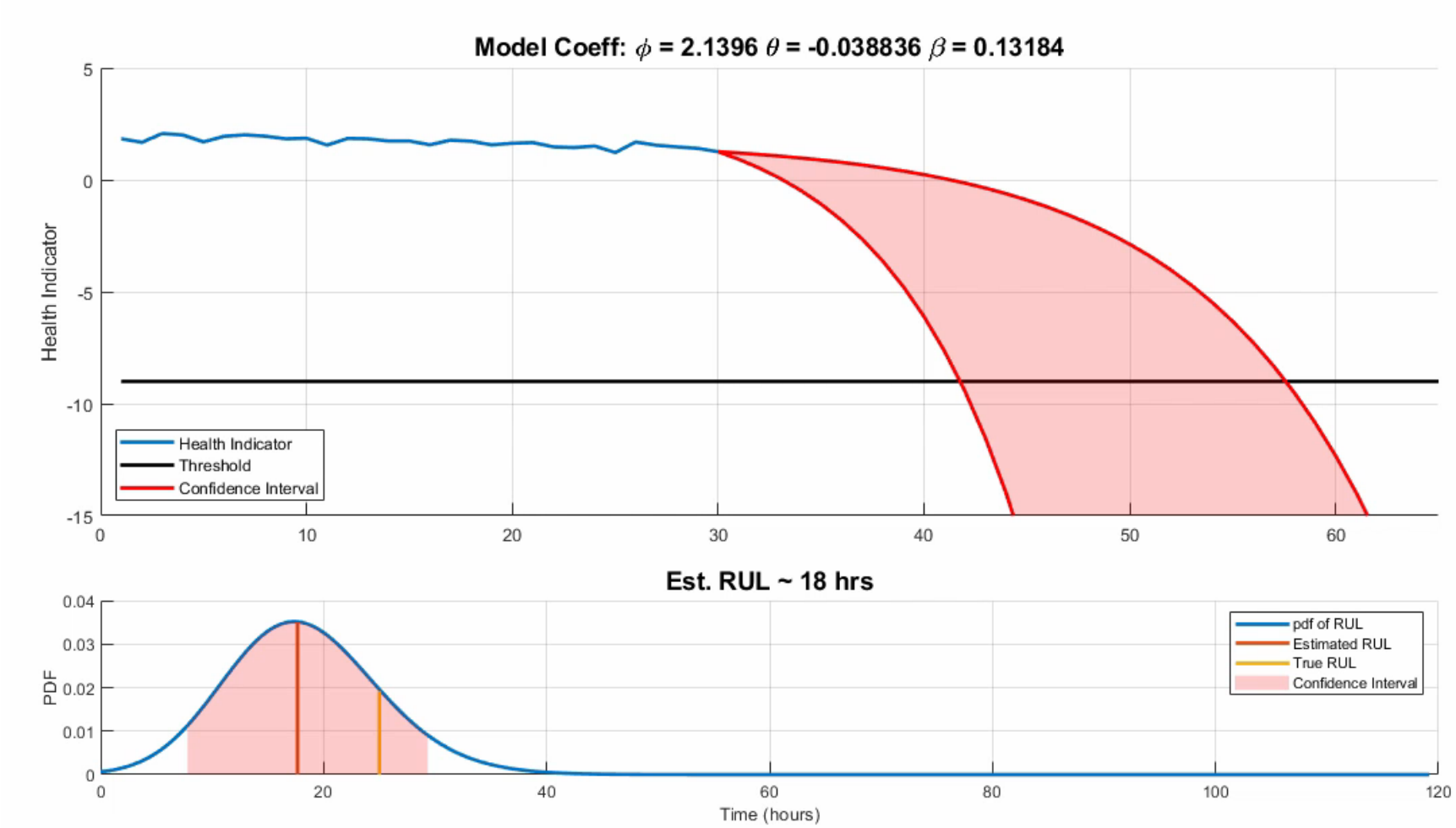
Develop Machine Learning Models



3 Develop Predictive Models

Process Engineer

Estimate Remaining Useful Life

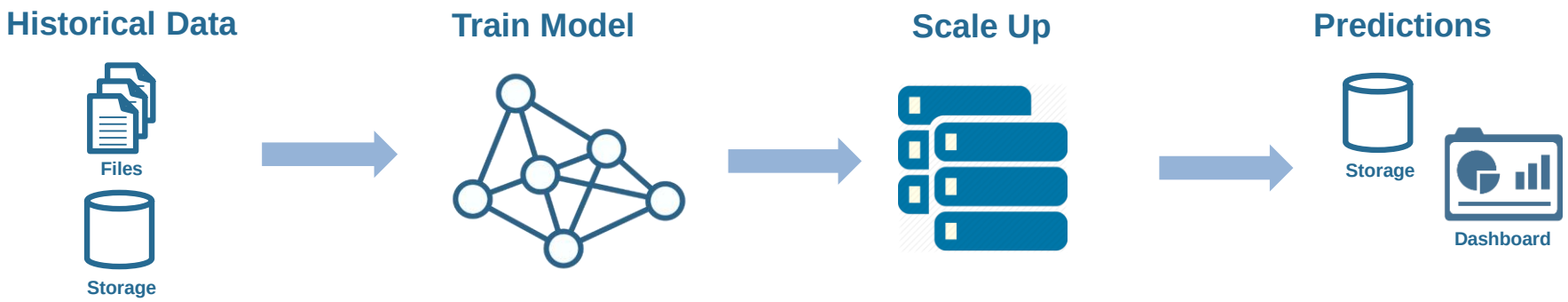


$$S(t) = \phi + \theta(t)\, e^{(\beta(t)t + \epsilon(t) - \frac{\sigma}{2})}$$

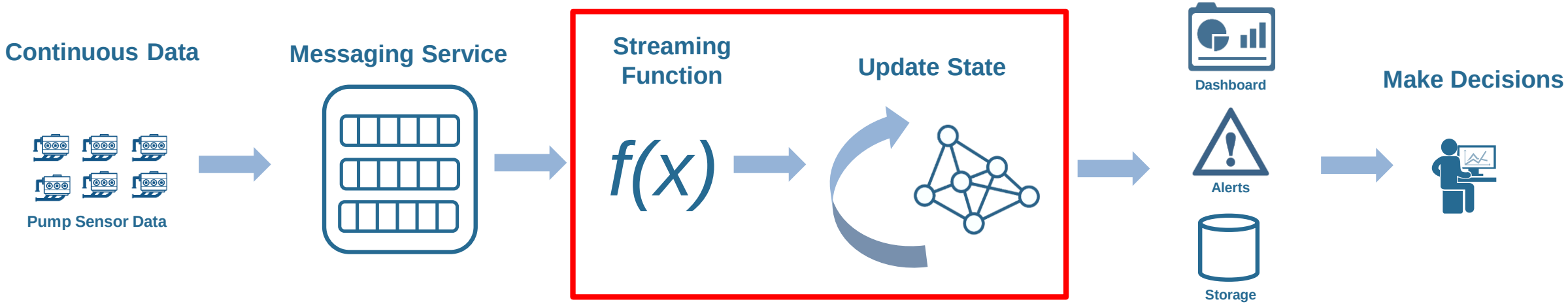
Develop a Stream Processing Function

Process Engineer

- ***Batch Processing:*** Build and test model on simulated data



- ***Stream Processing:*** Apply model to sensor data in near real-time



Process Engineer

4 Integrate with Production Systems

Develop a Stream Processing Function

Streaming Function

```
function new_state = streamingFunction(data,old_state)
```

Preprocess signals

```
[data,features] = preprocessData(data);
```

Predict faults

```
[Leak,Blocking,Bearing] = predictFaultValues(features);
FaultType = predictFault(features);
[RUL,Model] = predictUpdateRUL(data.Timestamp,data.Flow,500);
```

Update state

```
new_state = updateState(data,old_state);
```

Write results

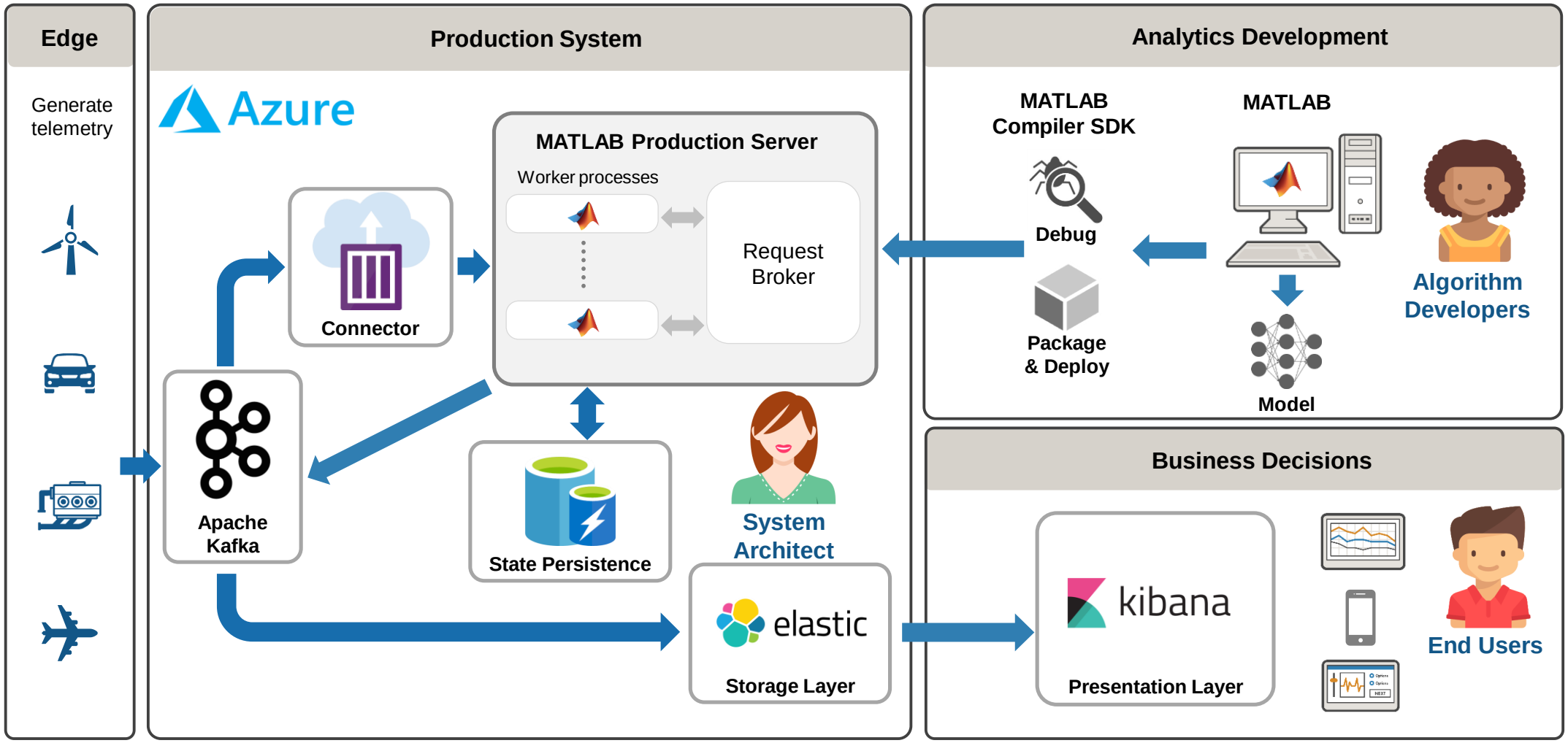
```
writeResults(Leak,Blocking,Bearing,FaultType,RUL,Model)
end
```

Process each window of data as it arrives

Previous state

Current window of data to be processed

Prototype Predictive Maintenance Architecture on Azure



What does a streaming function look like?

```
function pumpconsume(msg)
% Persistence service cache name must be unique to this group ID.
msg.metaInfo.cacheName = make_cache_name(msg.metaInfo.groupId);

% Application publishes results to a Kafka topic. The topic name
% uses the group ID to allow multiple simultaneous Kafka consumers --
% test and production, for example.
%
% consume and <consumerFunction> must agree on the name of the metaInfo
% field used to store the topic name.
msg.metaInfo.resultTopic = strcat('pump_results_', ...
    matlab.lang.makeValidName(msg.metaInfo.groupId));

% The name of the connection to the persistence service. Configured
% (set) by MATLAB Production Server admin.
msg.metaInfo.connection = 'Azure_Redis';

% Variables in the data table that we publish to Kafka.
msg.metaInfo.outputVariables = ...
    struct('Flow',NaN,'Pressure',NaN,'Current',NaN);

% Emit more log messages.
msg.metaInfo.Verbose = true;

% Finally, consume (process) the data in thie input message.
mps.stream.consume(msg);
end
```

Topic

Persistence

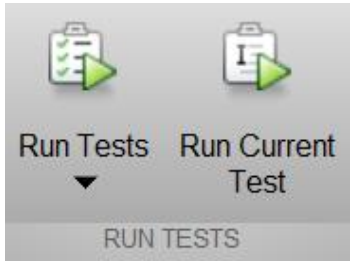
Expected Signals

Consume

Process Engineer

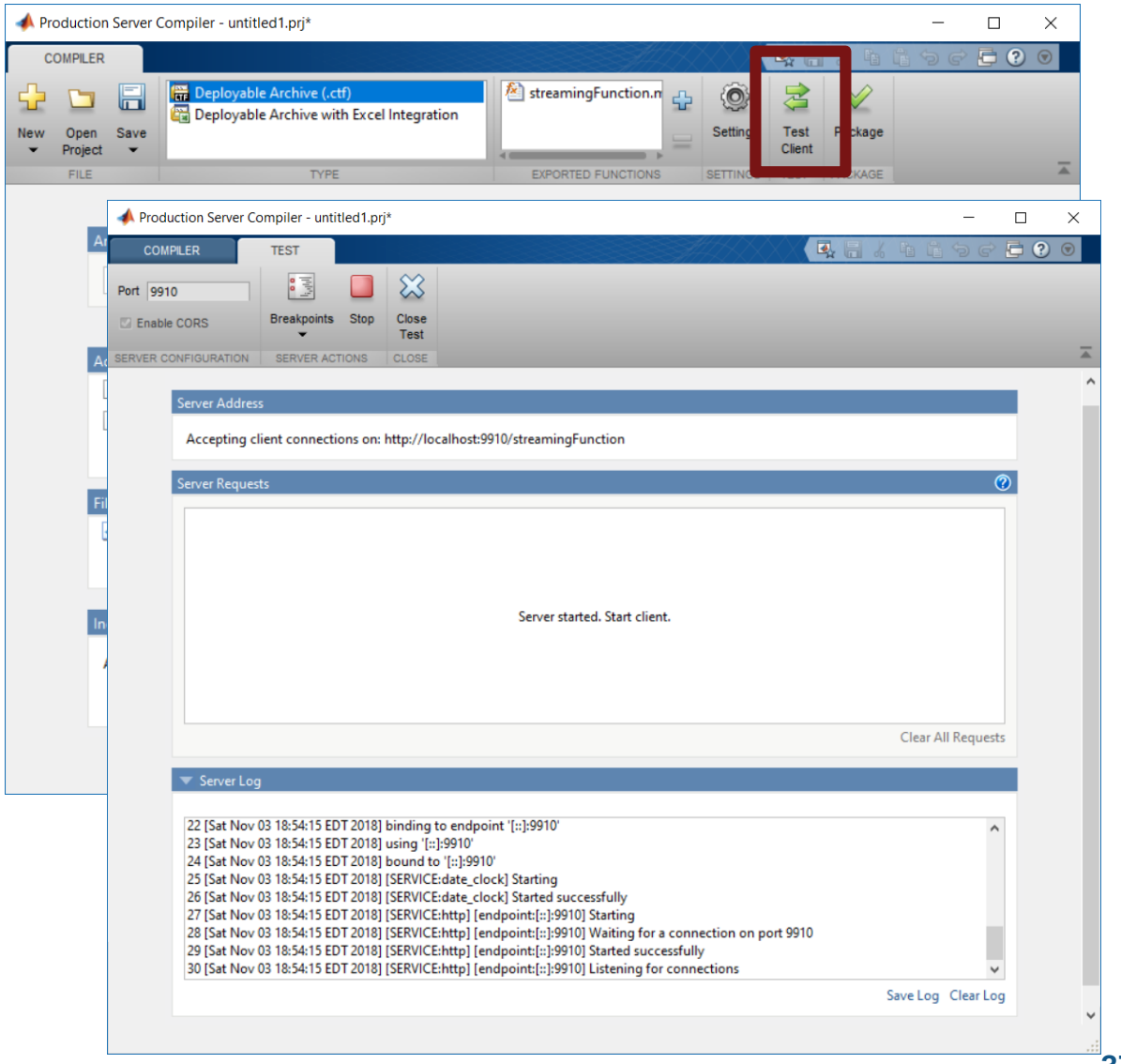
4 Integrate with Production Systems

Test Stream Processing Function

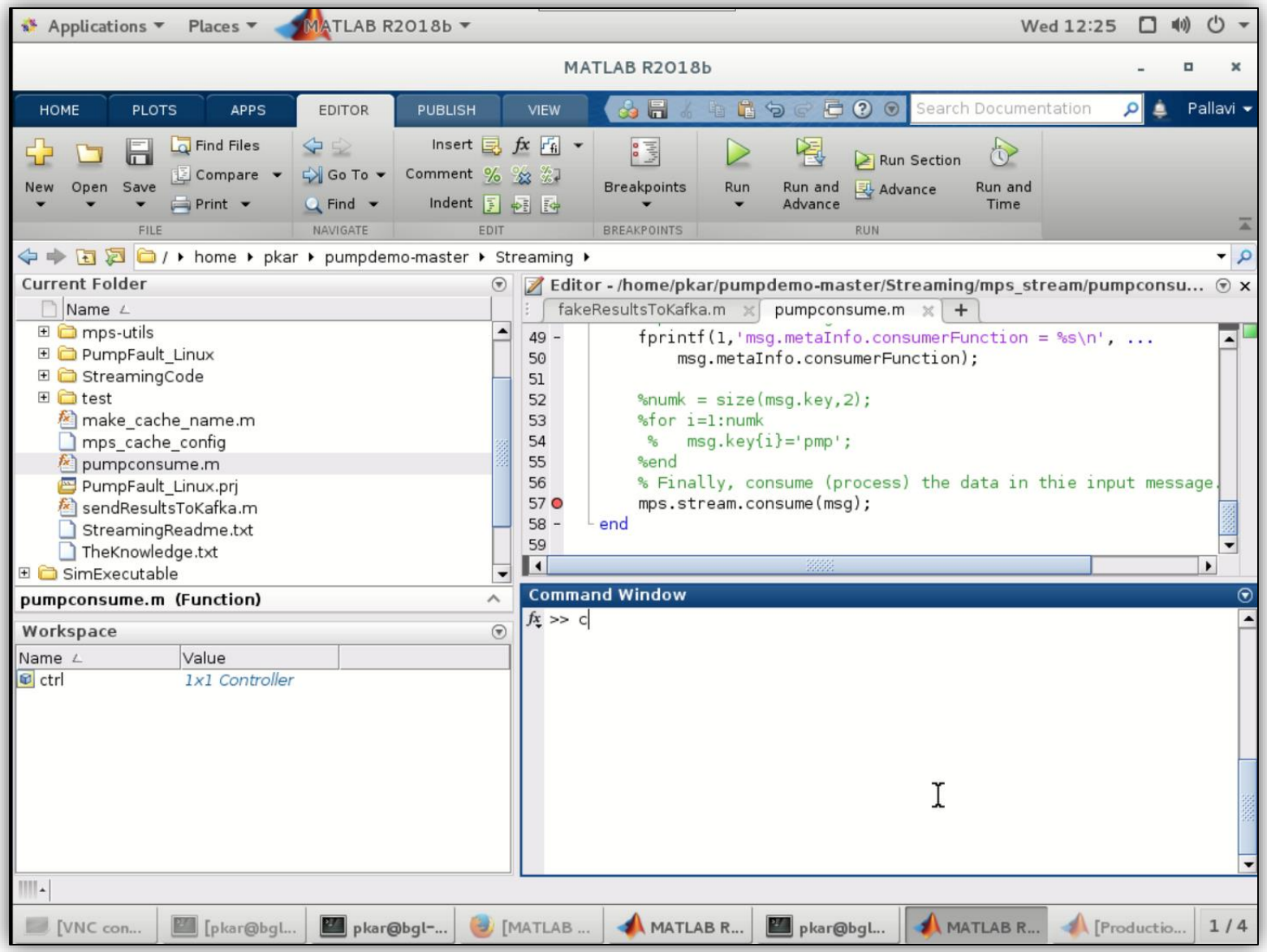



```
results = runtests('predictFaults_tests')
```

```
Running predictFaults_tests
....
Done predictFaults_tests
__________
results =
  1×4 TestResult array with properties:

    Name
    Passed
    Failed
    Incomplete
    Duration
    Details
Totals:
   4 Passed, 0 Failed, 0 Incomplete.
   0.01614 seconds testing time.
```



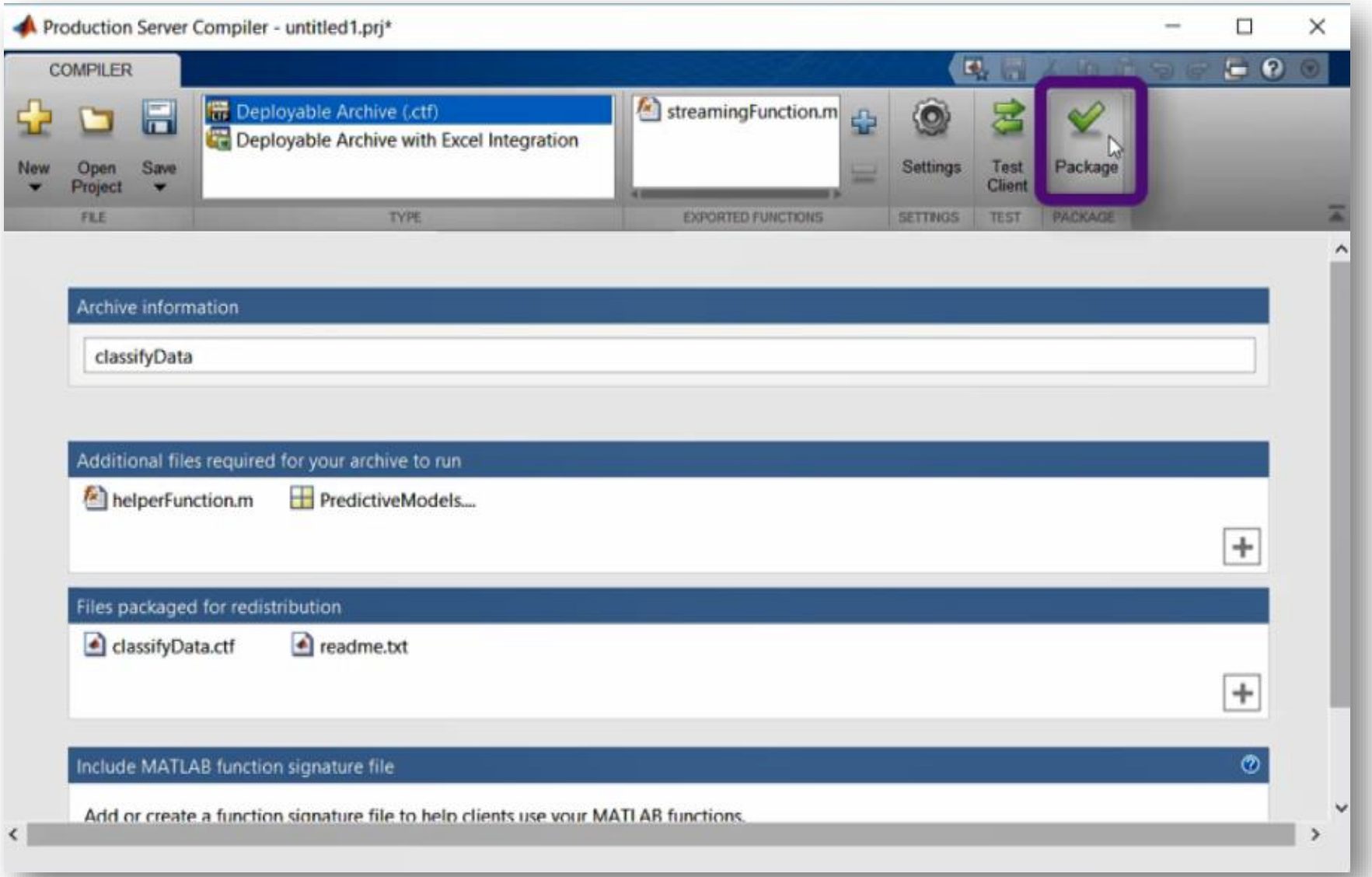
Test and Debug Streaming Function



Process Engineer

4 Integrate with Production Systems

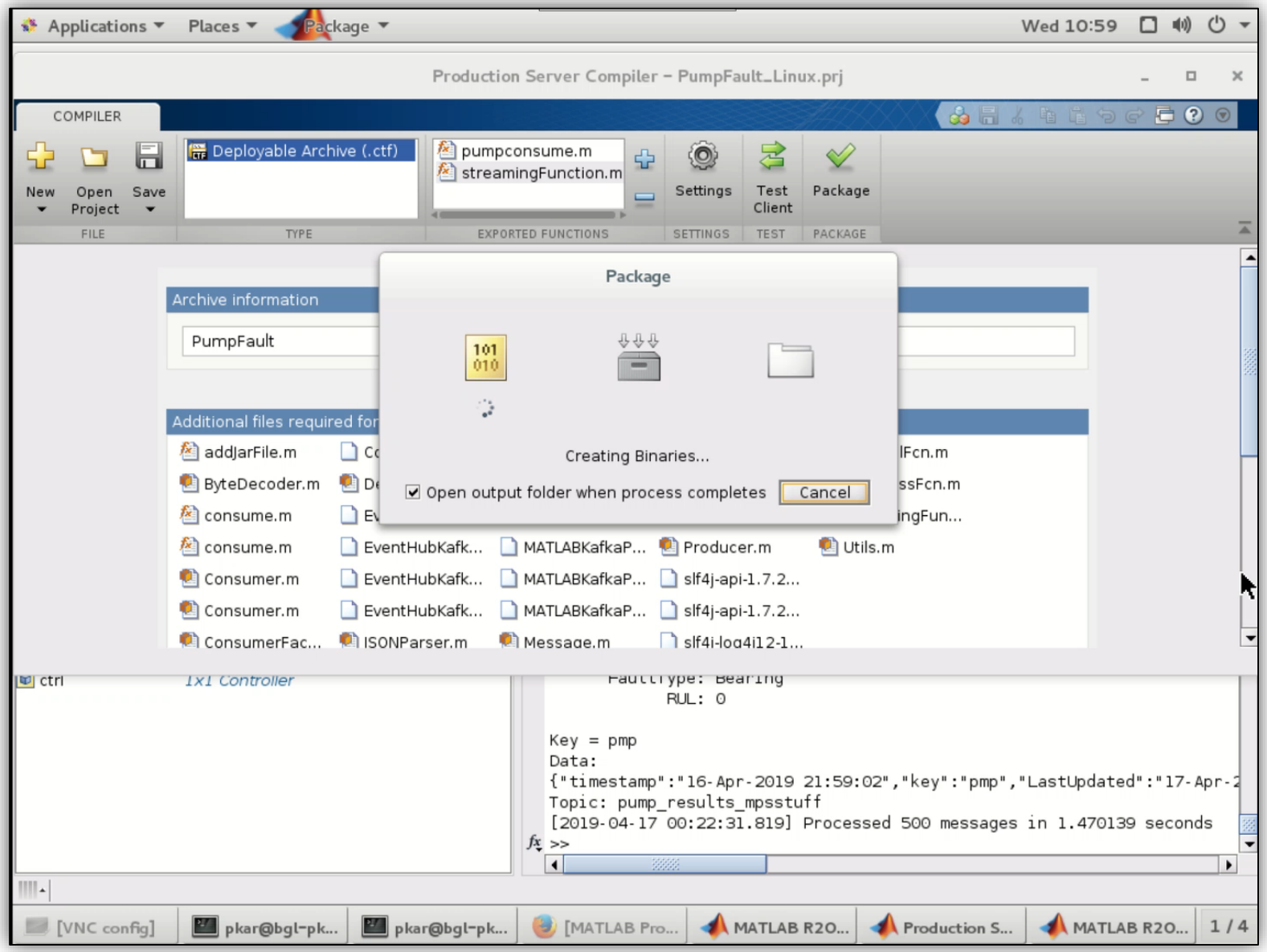
Package Stream Processing Function



Process Engineer

4 Integrate with Production Systems

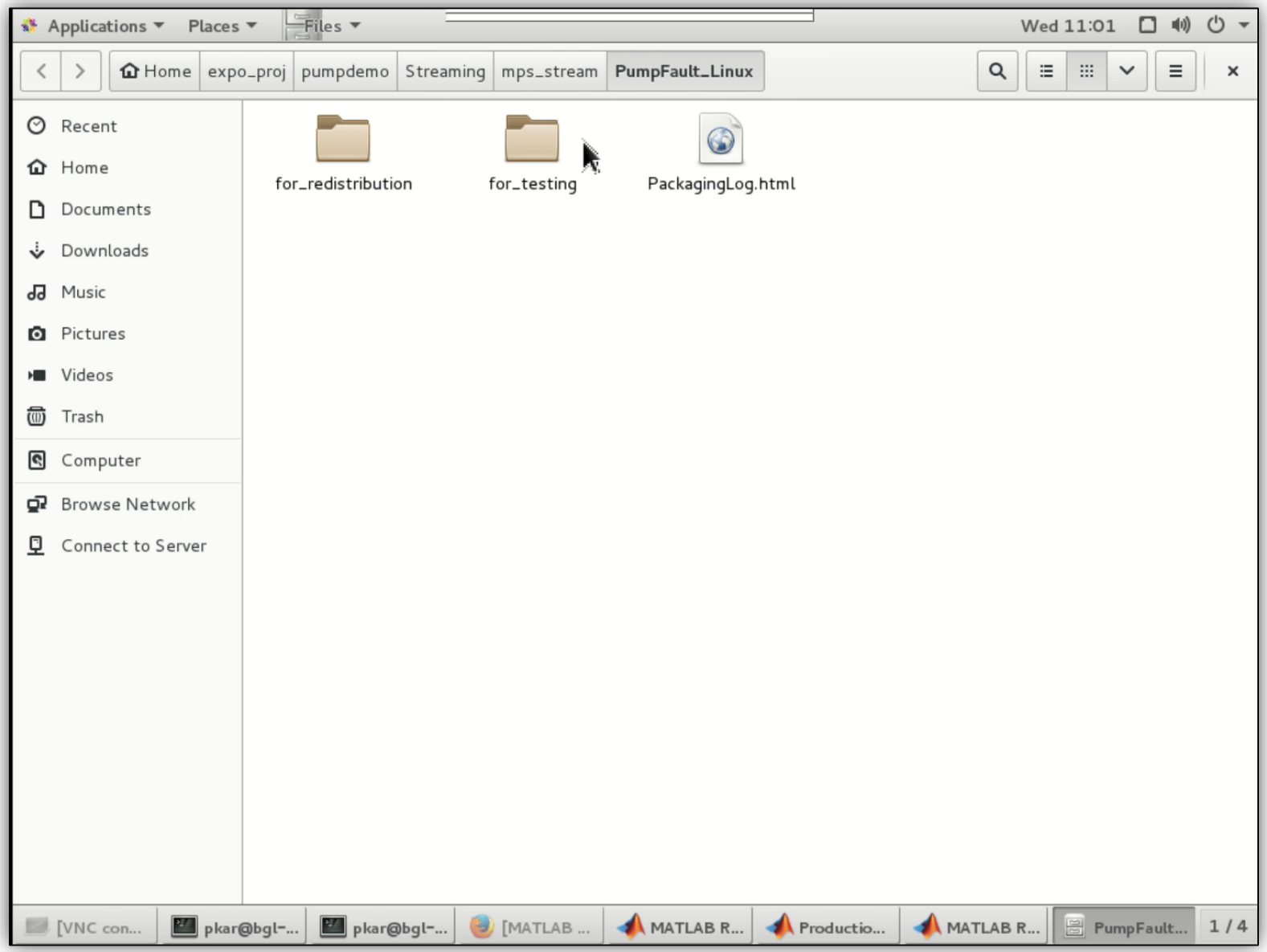
Package and Test to generate compiled archive



Process Engineer

4 Integrate with Production Systems

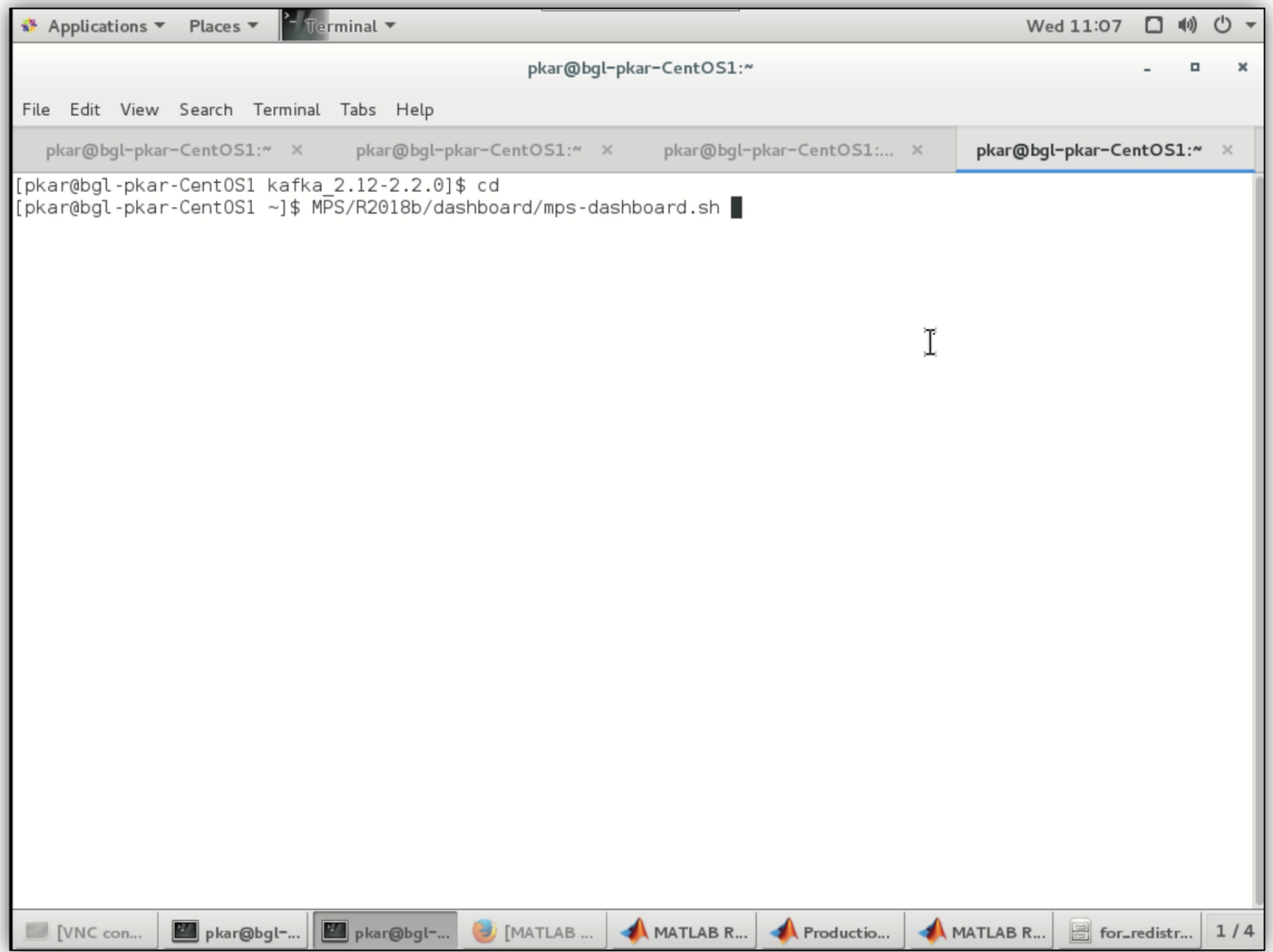
Compiled Package and Runtime requirements



Process Engineer

4 Integrate with Production Systems

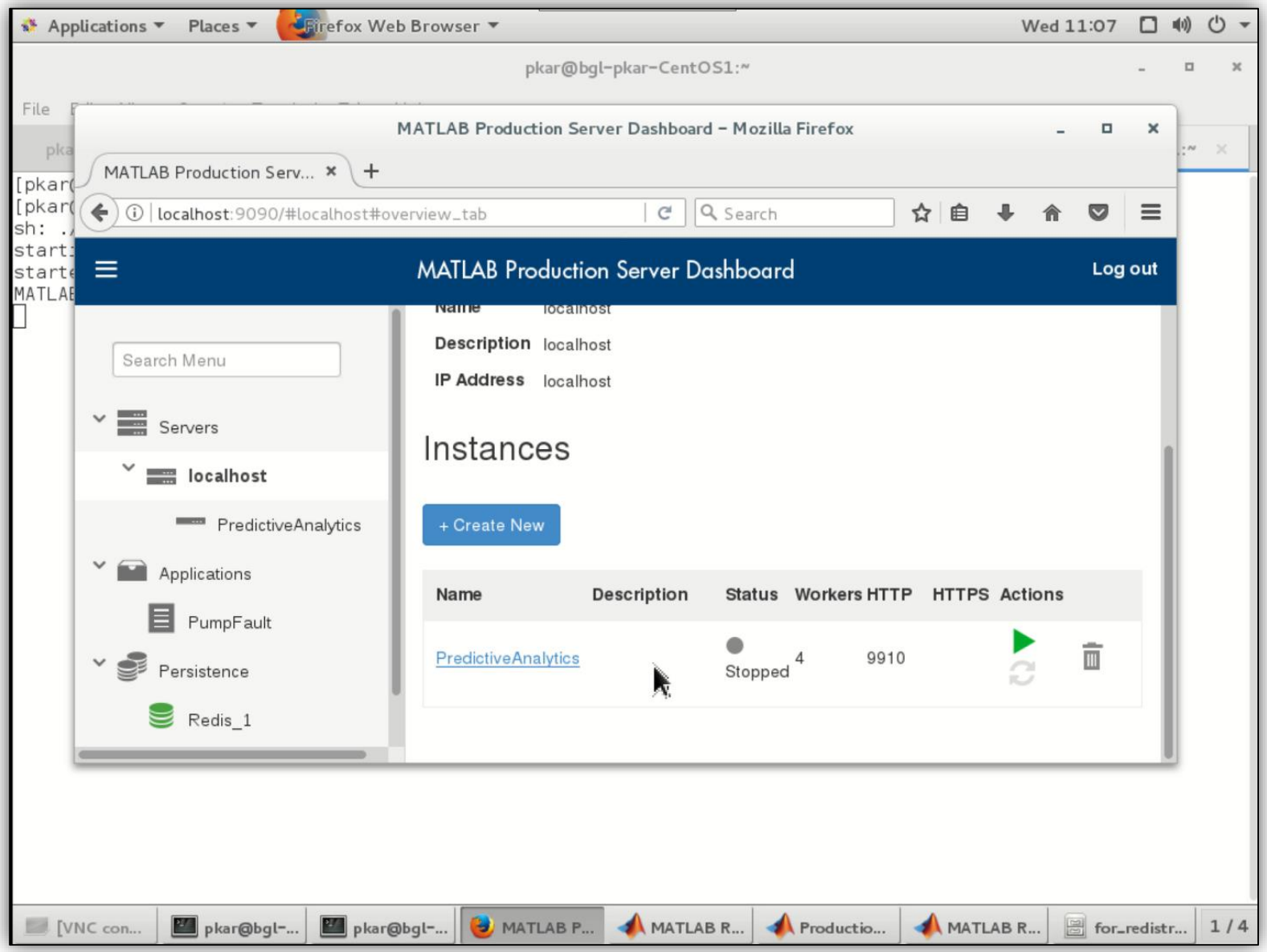
Starting MATLAB Production Server Dashboard

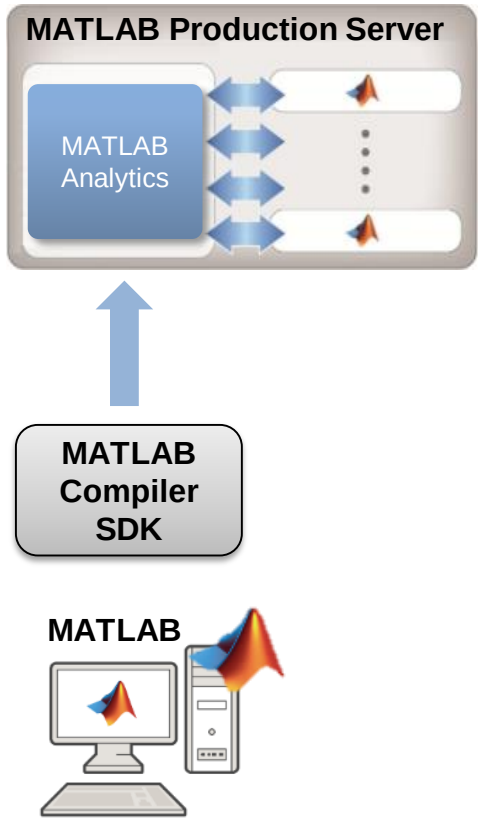


Process Engineer

4 Integrate with Production Systems

Deploying Streaming Function on Production System

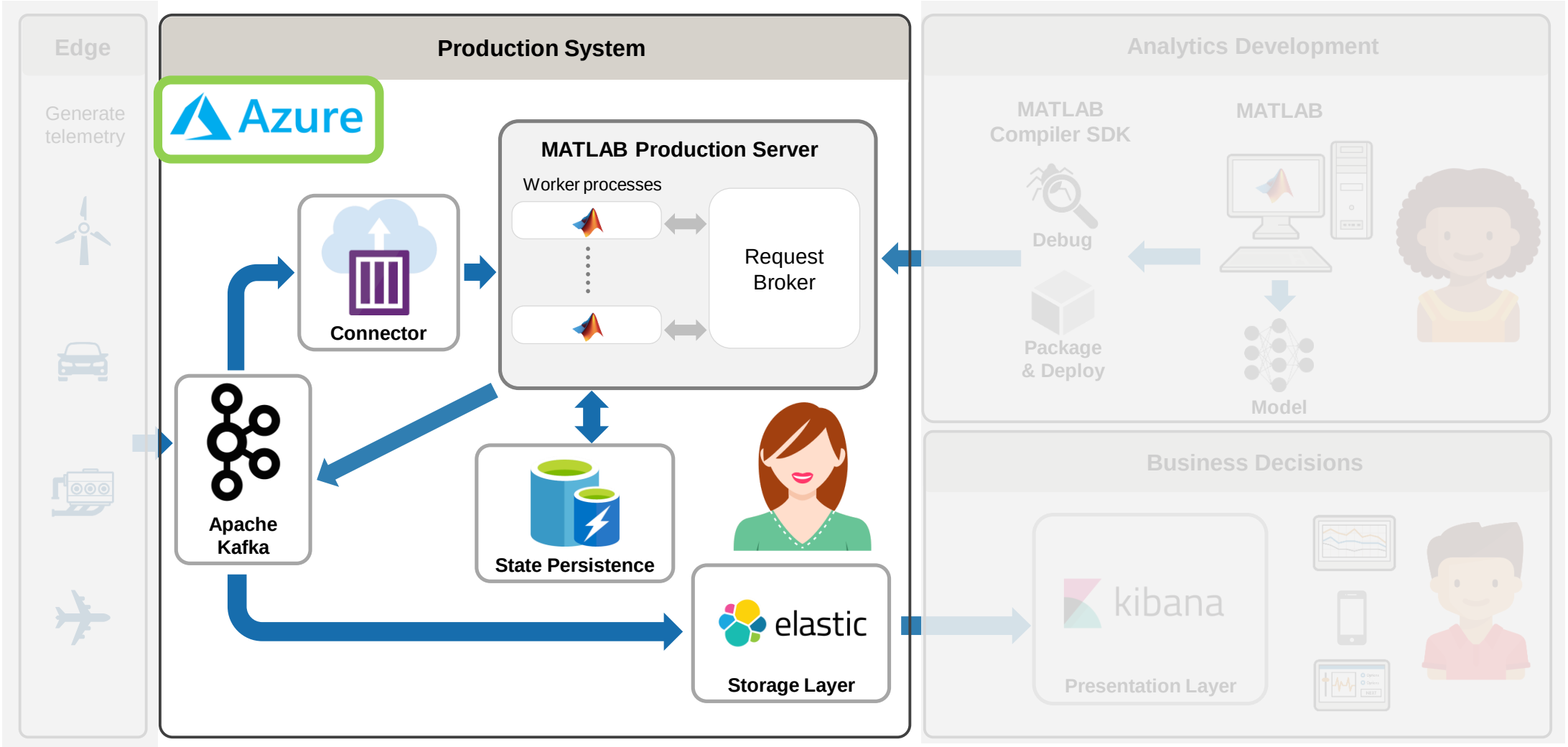




System Architect

4 Integrate with Production Systems

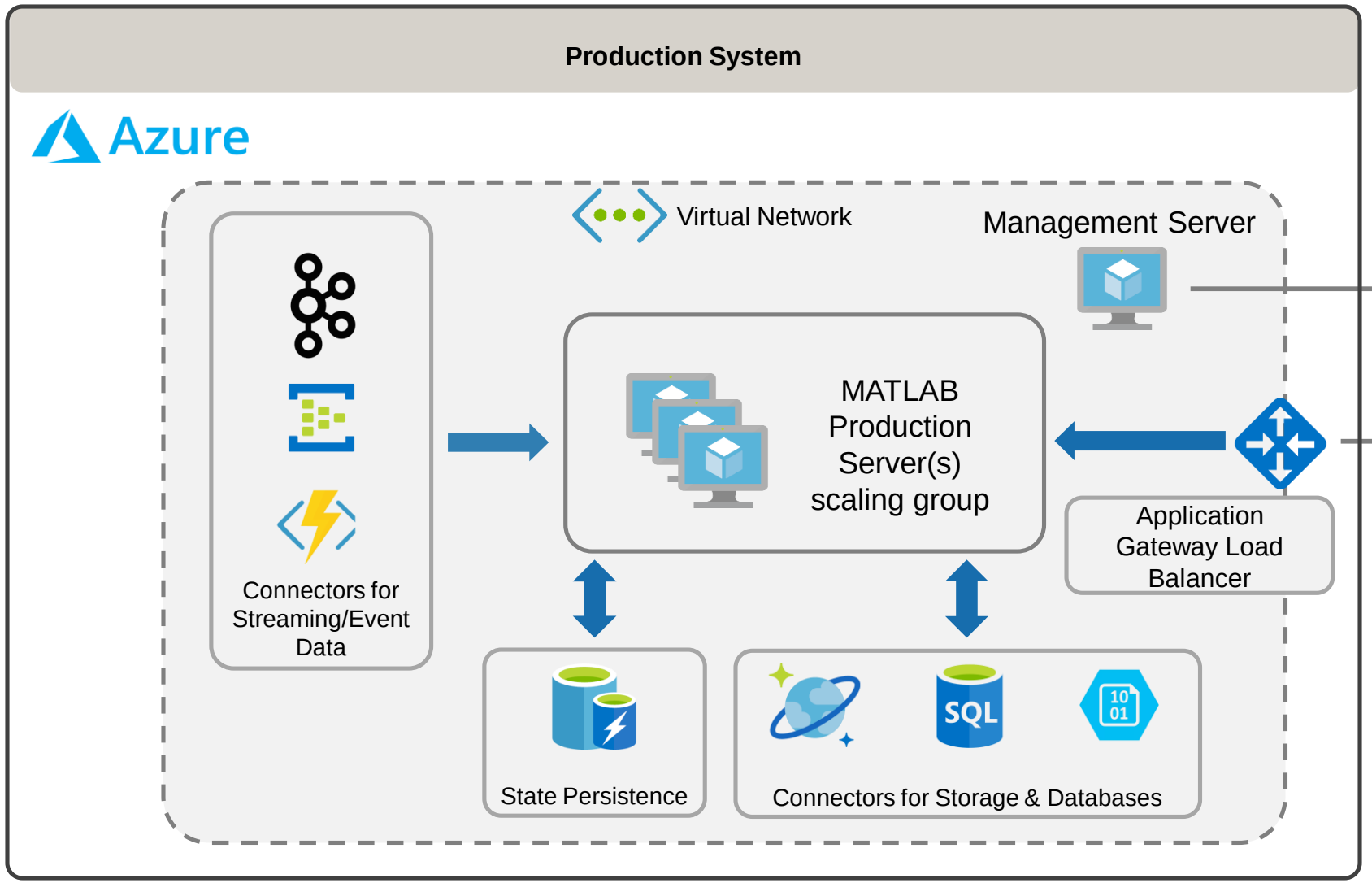
Integrate Analytics with Production Systems



System Architect

4 Integrate with Production Systems

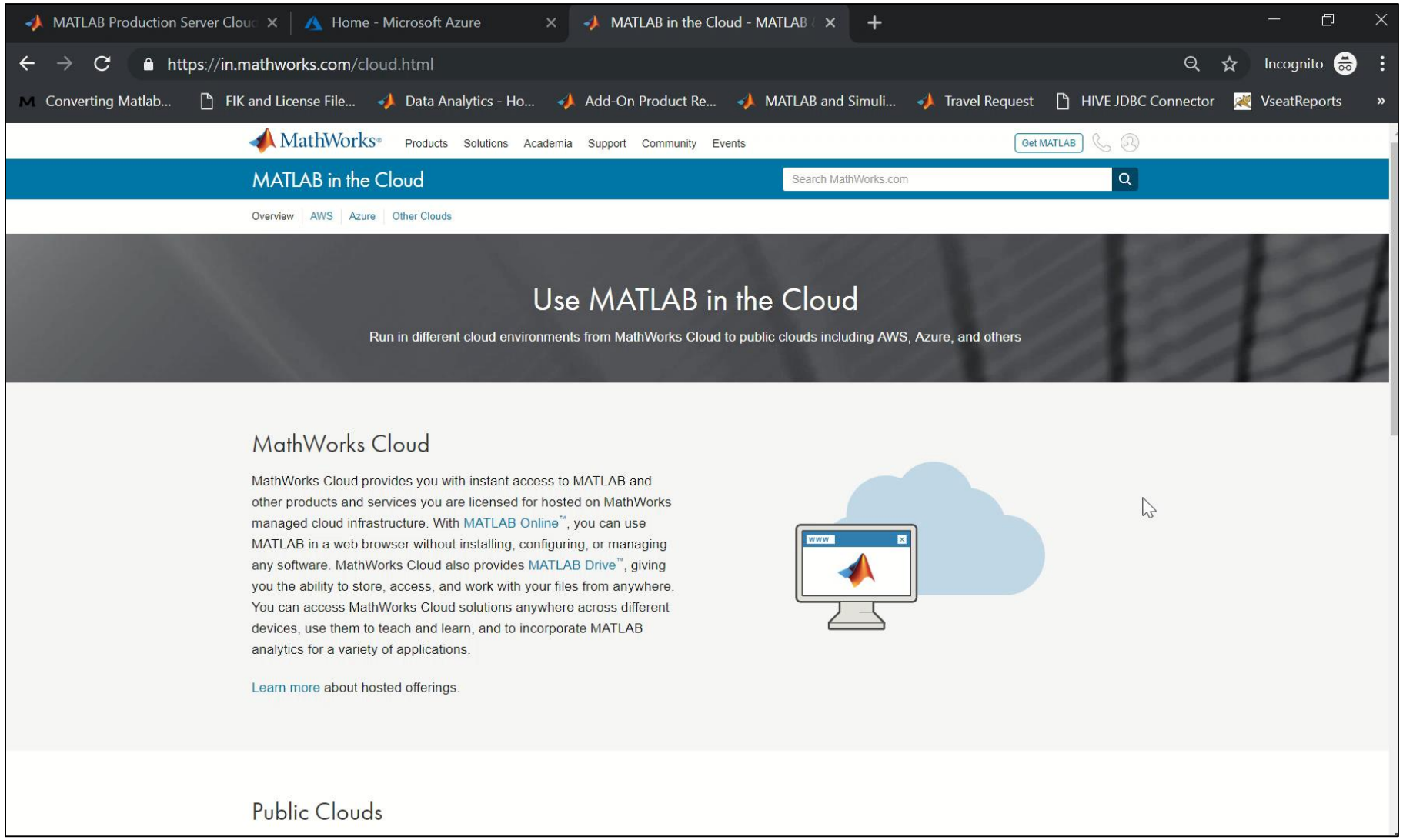
MATLAB Production Server on Azure



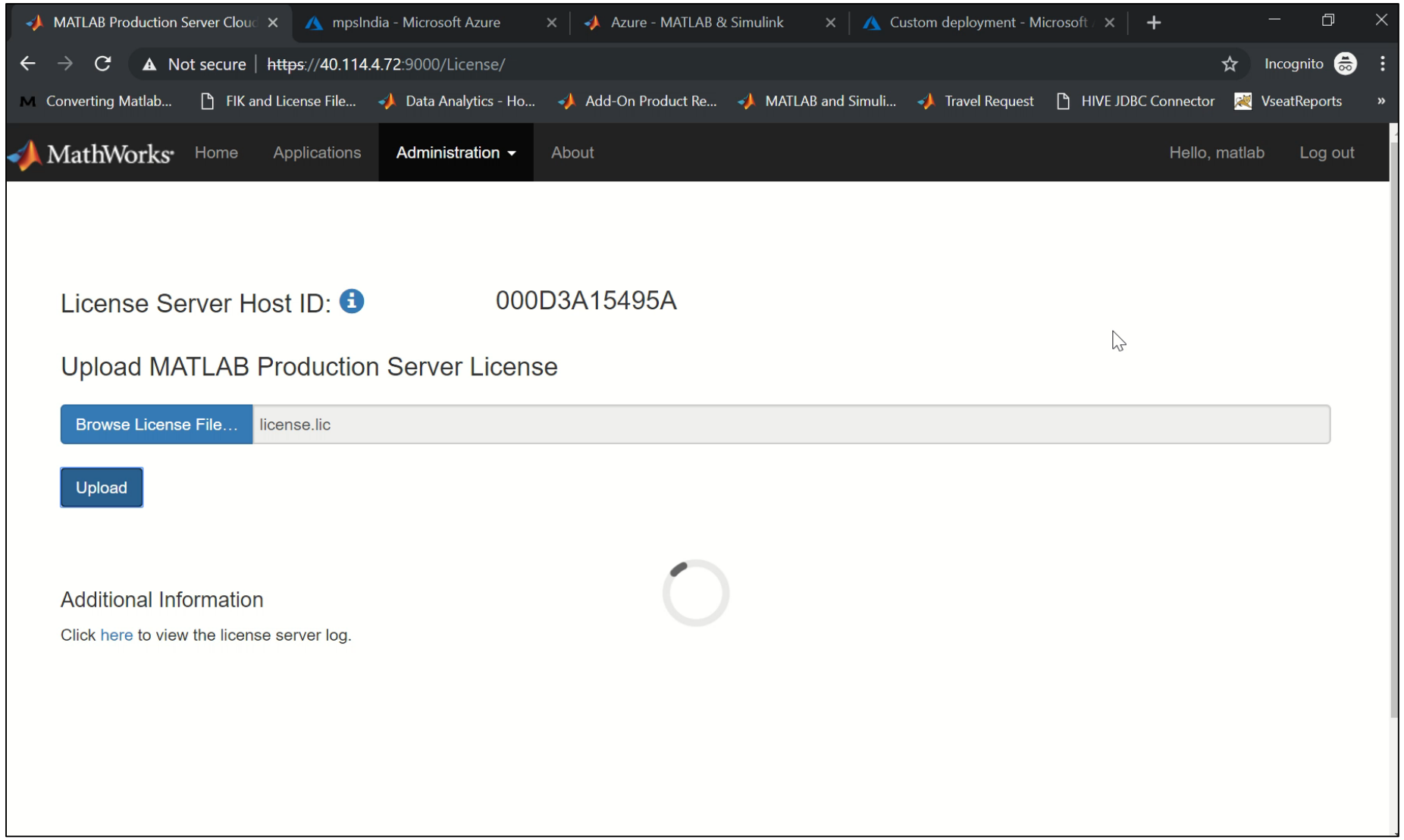
https management endpoint

Enterprise Applications

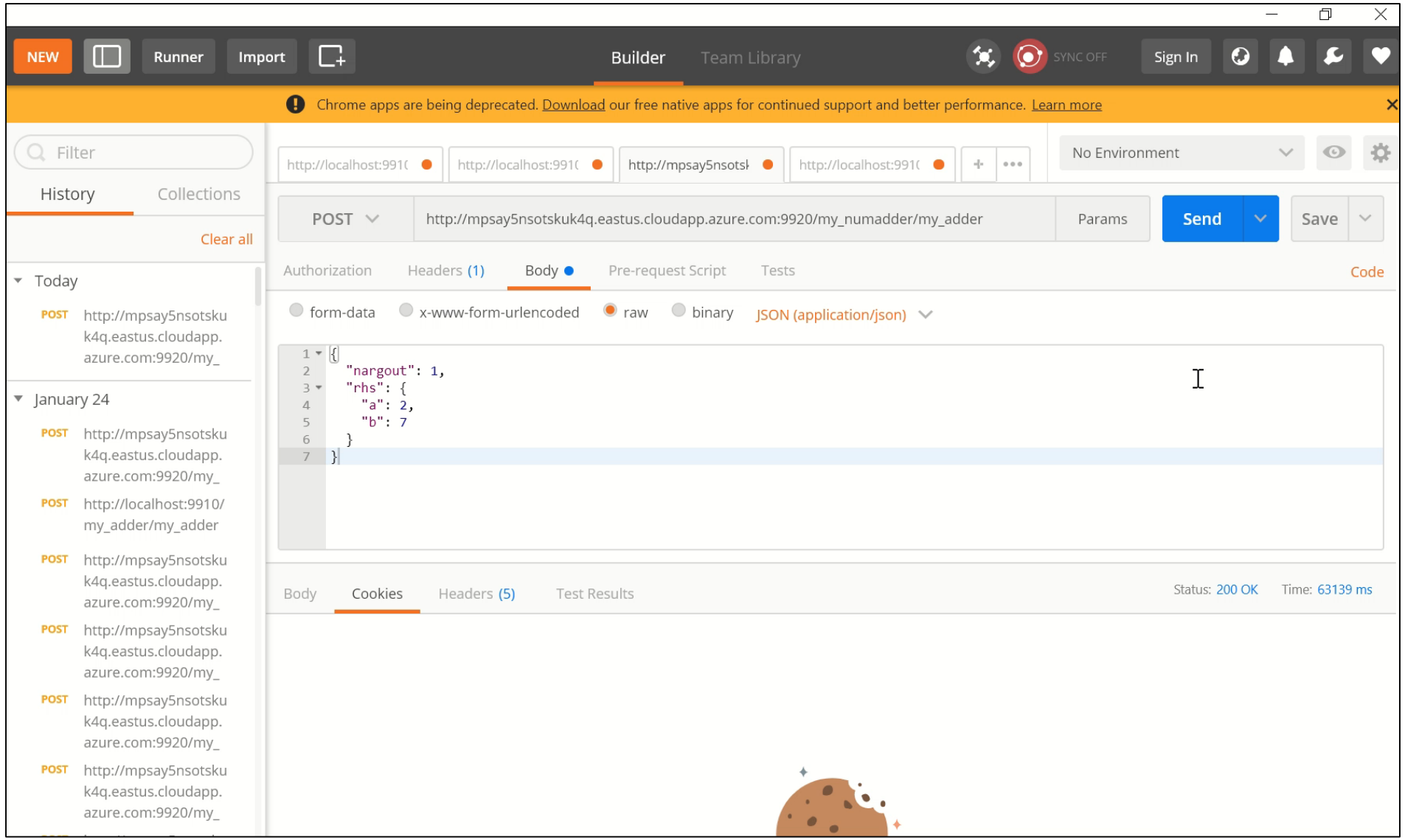
MathWorks Cloud Reference Architecture



MPS License and Instance Settings



Serving REST Calls on Production Server



MathWorks Reference Architectures

mathworks.github.io

https://mathworks.com/cloud Verified

matlab-aws-s3
MATLAB interface for AWS S3.

MATLAB Updated 26 days ago

matlab-azure-blob
MATLAB interface for Windows Azure Blob Storage.

MATLAB Updated on Feb 21

matlab-parquet
MATLAB Interface for Apache Parquet

MATLAB ★ 1 Updated on Dec 20, 2018

matlab-azure-data-lake
MATLAB Interface for Azure Data Lake.

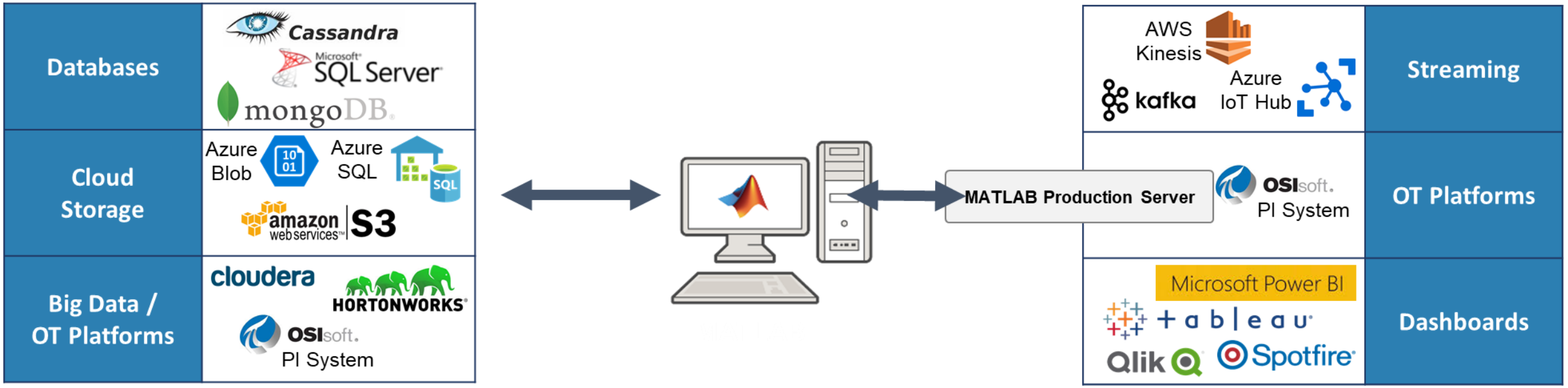
MATLAB Updated on Feb 21

matlab-aws-common
Code common to MATLAB interfaces. Code in this repository is used as a dependency for other projects such as matlab-aws-s3.

MATLAB Updated on Feb 21

matlab-avro
MATLAB interface for Apache Avro files.

MATLAB Updated on Feb 9



System Architect

4 Integrate with Production Systems

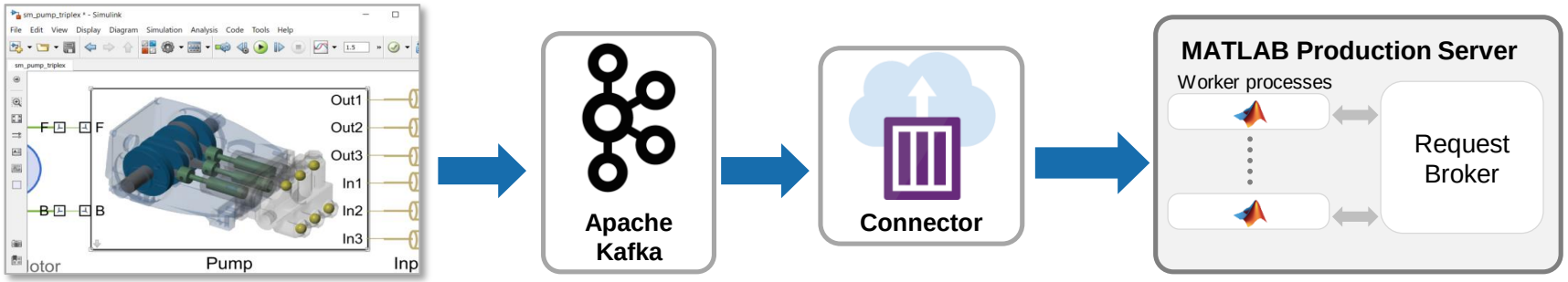
Review System Requirements

- Requirements from the Process Engineer
  - Every millisecond, each pump generates a time-stamped record of flow, pressure, and current
  - Model expects 1 sec. window of data per pump
  - Initially, 1's – 10's of devices, but quickly scale to 100's

Process Engineer

- Requirements from the Operator
  - Alerts when parameters drift outside the expected ranges
  - Continuous estimating of RUL for each pump

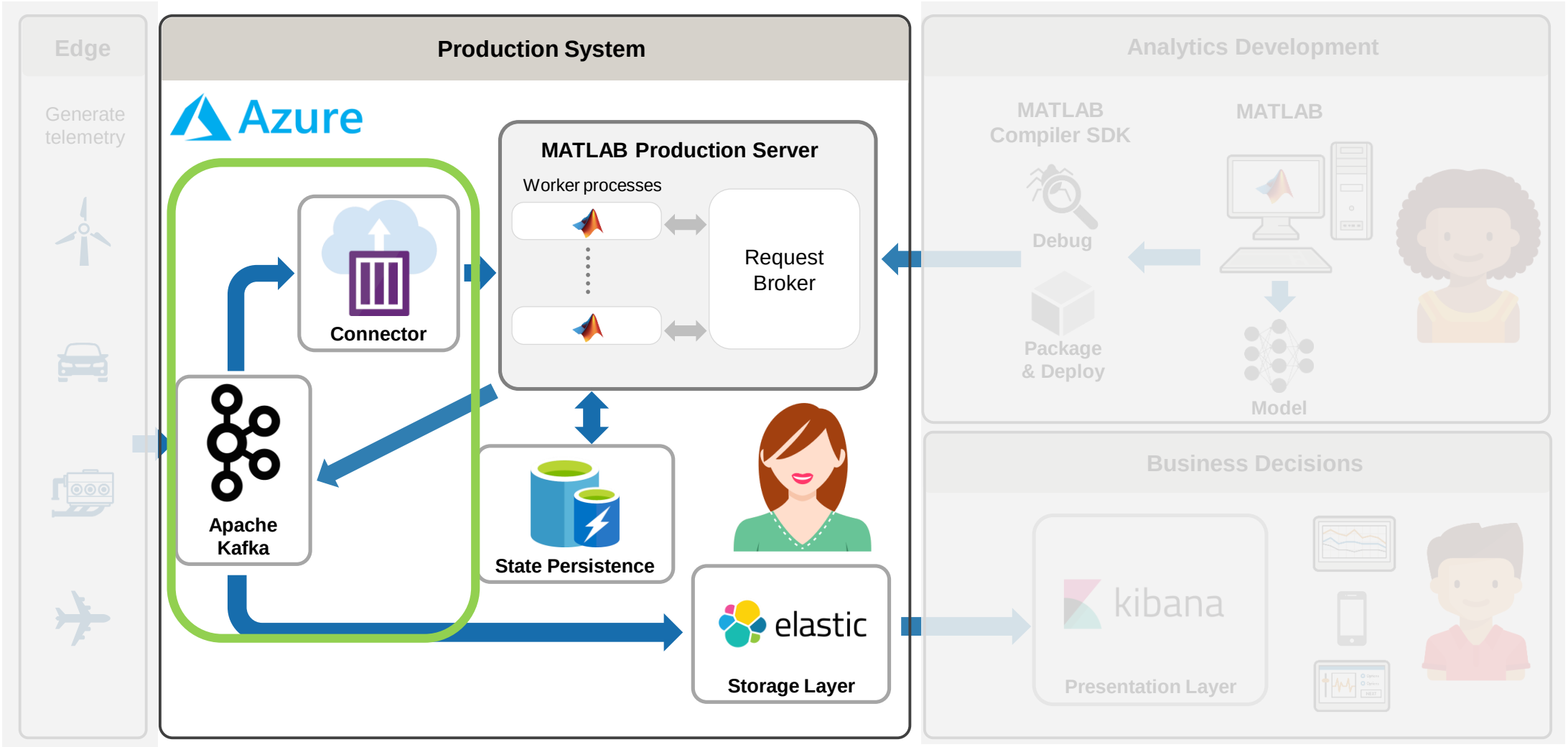
Operator



System Architect

4 Integrate with Production Systems

Integrate Analytics with Production Systems

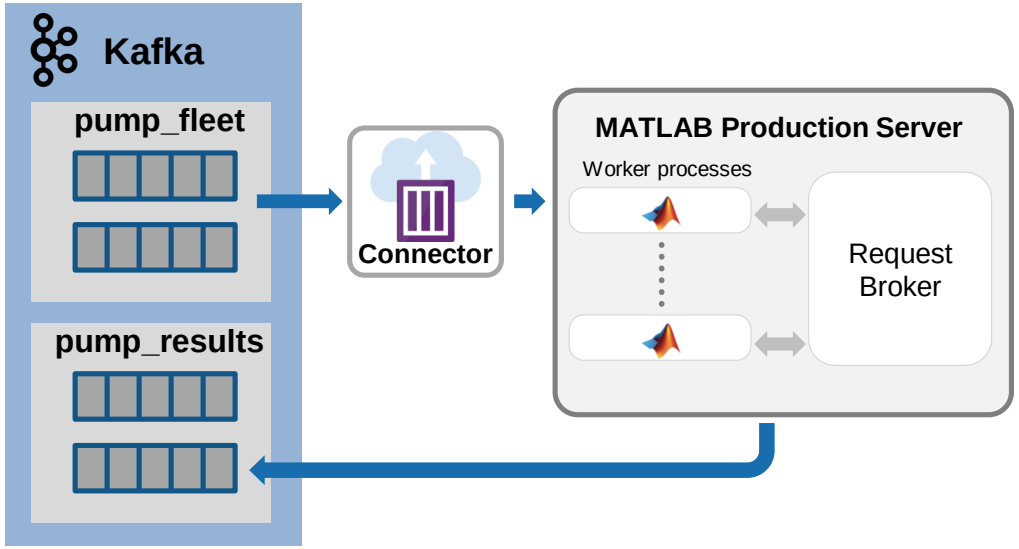


System Architect

4 Integrate with Production Systems

Connecting MATLAB Production Server to Kafka

- Connector feeds single **Kafka topic to** a MATLAB function
- **Publisher library** for MATLAB for writing to a results stream
- Connector Features:
  - Deploy as a micro-service with Docker
  - Drive everything through config
  - Group data into time windows and pass to MATLAB as a timetable
  - Use Kafka's check-pointing (i.e. at-least-once)



Setting up the Kafka Connector

```
#!/usr/bin/env bash

BASE=$(pwd)

echo "BASE: ${BASE}"

MPS_HOME=$BASE
KAFKA_CONNECTOR_DIR=${MPS_HOME}/../../kafka-connector/Software/Java
MPS_CLIENT=${KAFKA_CONNECTOR_DIR}/client/java/mps_client.jar
KAFKA_CONNECTOR=${KAFKA_CONNECTOR_DIR}/lib/com/mathworks/mps/kafka-connector/1.1.0/kafka-connector-1.1.0-jar-wi$
CLASSPATH=${MPS_CLIENT}:${KAFKA_CONNECTOR}

# CLASSPATH=${CLASSPATH}:${MPS_HOME}/lib/com/mathworks/mps/kafka-connector/1.1.0/kafka-connector-jar-with-depen$

export GROUP_ID=mpsstuff
export CONNECTOR_TOPIC=to-mps
export CONNECTOR_TOPIC_OUT=from-mps
export MPS_CONNECT=http://localhost:9910
export MPS_ARCHIVE=PumpFault
export MPS_FUNCTION=streamingFunction
export BOOTSTRAP_SERVERS=localhost:9092
export MPS_DISPATCH_FUNCTION=pumpconsume
# Skip or stop
export MATLAB_ERROR_ACTION=skip

echo "CLASSPATH == ${CLASSPATH}"

exec java -cp ${CLASSPATH} \
  -Dlog4j.configuration=file:${MPS_HOME}/config/log4j.properties \
  com.mathworks.mps.client.kafka.KafkaConnector $@
```

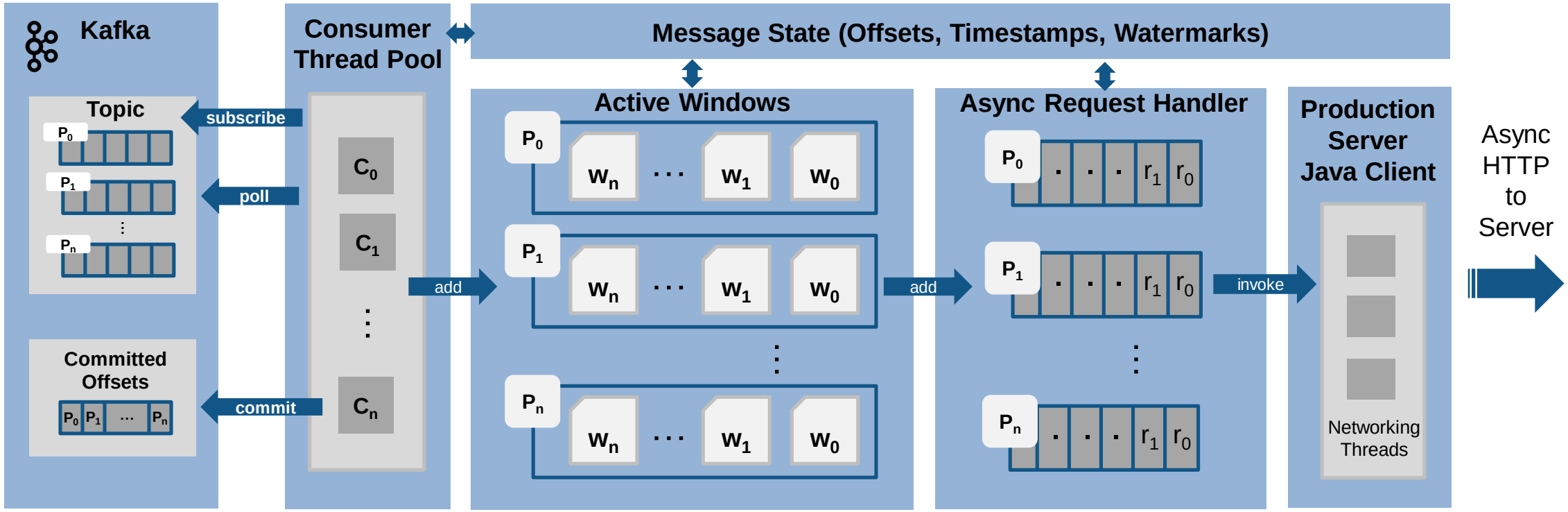
Libraries

Configuration

System Architect

4 Integrate with Production Systems

Kafka connector architecture



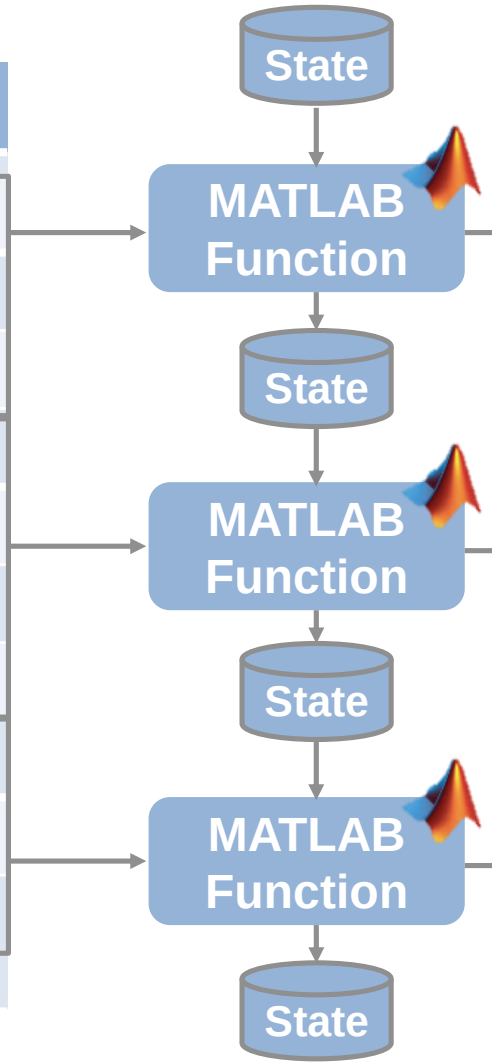
System Architect

4 Integrate with Production Systems

Streaming data is treated as an unbounded Timetable

Input Stream

Event Time	Pump Id	Flow	Pressure	Current
18:01:10	Pump1	1975	100	110
18:10:30	Pump3	2000	109	115
18:05:20	Pump1	1980	105	105
18:10:45	Pump2	2100	110	100
18:30:10	Pump4	2000	100	110
18:35:20	Pump4	1960	103	105
18:20:40	Pump3	1970	112	104
18:39:30	Pump4	2100	105	110
18:30:00	Pump3	1980	110	113
18:30:50	Pump3	2000	100	110
…	…	…	…	…



Output Stream

Time window		Pump Id	Bearing Friction
…		…	…
18:00:00	18:10:00	Pump1	5
		Pump3	…
		Pump4	…
18:10:00	18:20:00	Pump2	7
		Pump3	3
		Pump4	…
18:20:00	18:30:00	Pump1	…
		Pump3	4
		Pump4	…
18:30:00	18:40:00	Pump5	…
		Pump3	5
		Pump4	8

System Architect

4 Integrate with Production Systems

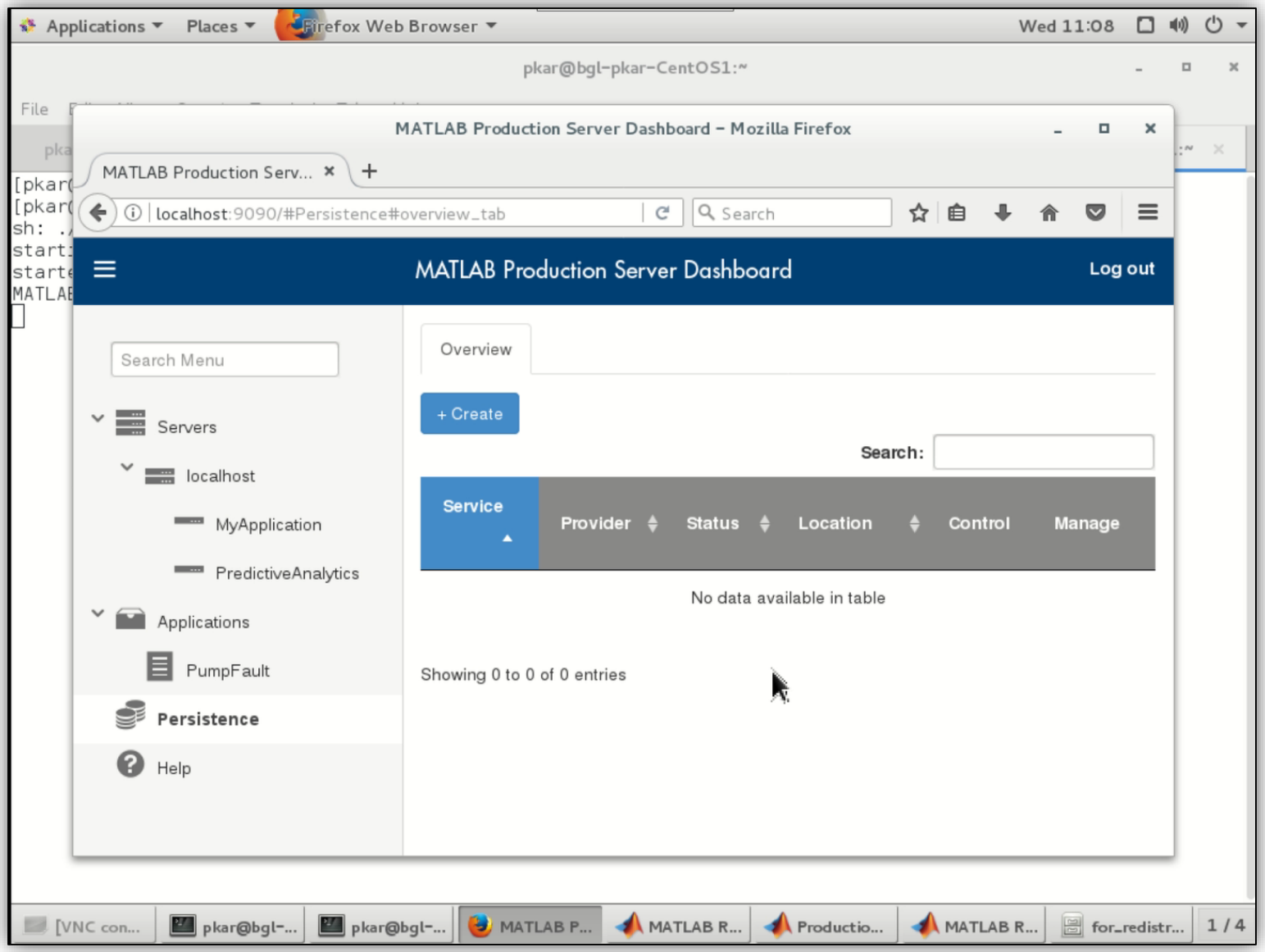
Messaging adapter for Production Server

- Bridges streaming data and Production Server Async Java Client
- Batches incoming messages and sends them via HTTP request/response
  - Time windows, event time processing, and out-of-order data
- Uses Asynchronous pipeline model with back-pressure
  - Kafka consumers are automatically paused when server is busy
- Supports sequential (stateful) and unordered (stateless) processing
  - Provide unique stream ID/topic/partition info for persistence layer
- Pass data as MATLAB timetables
- Partition aware – enables full exploitation of partition-based parallelism

Process Engineer

4 Integrate with Production Systems

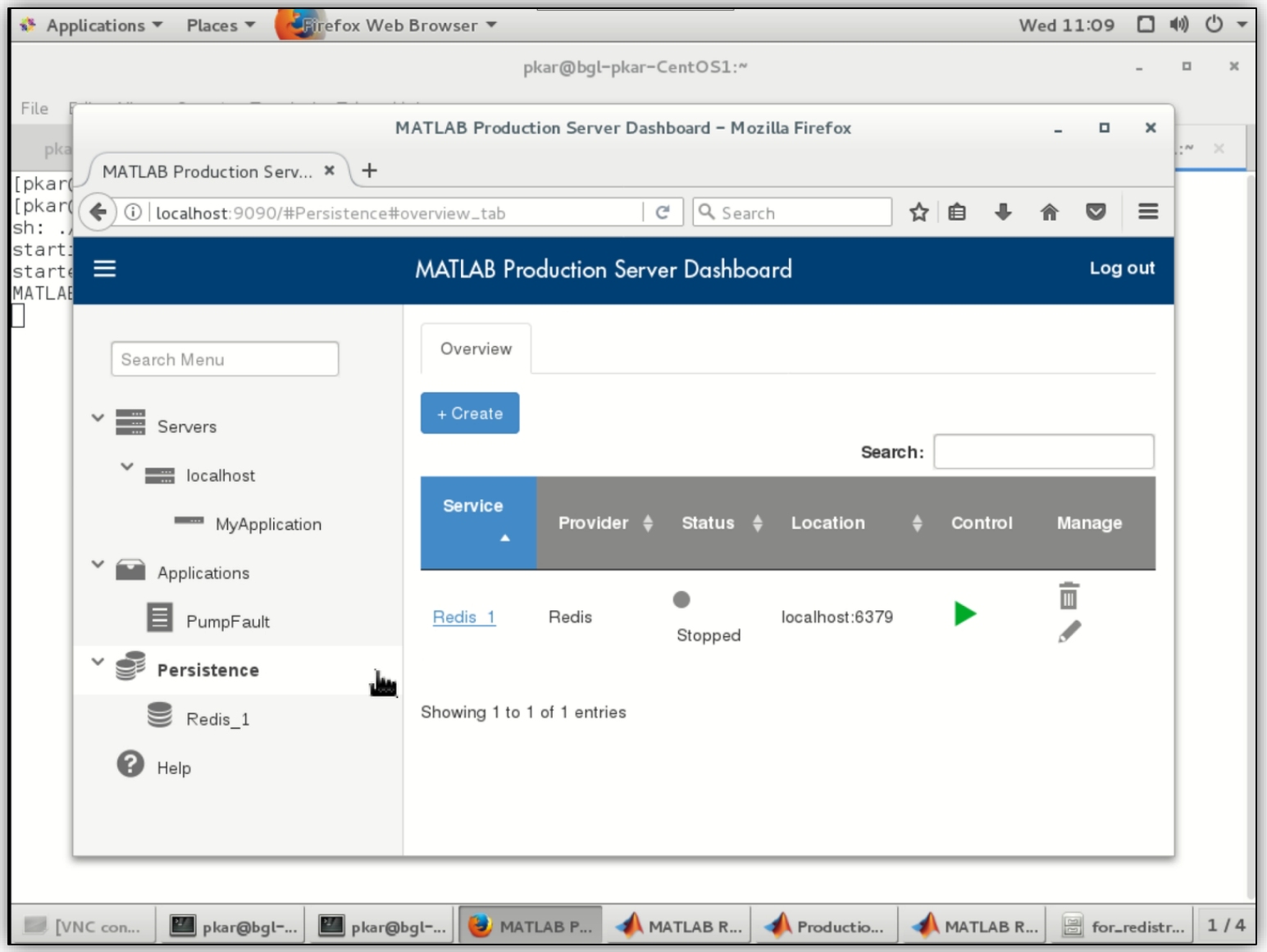
Creating persistence



Process Engineer

4 Integrate with Production Systems

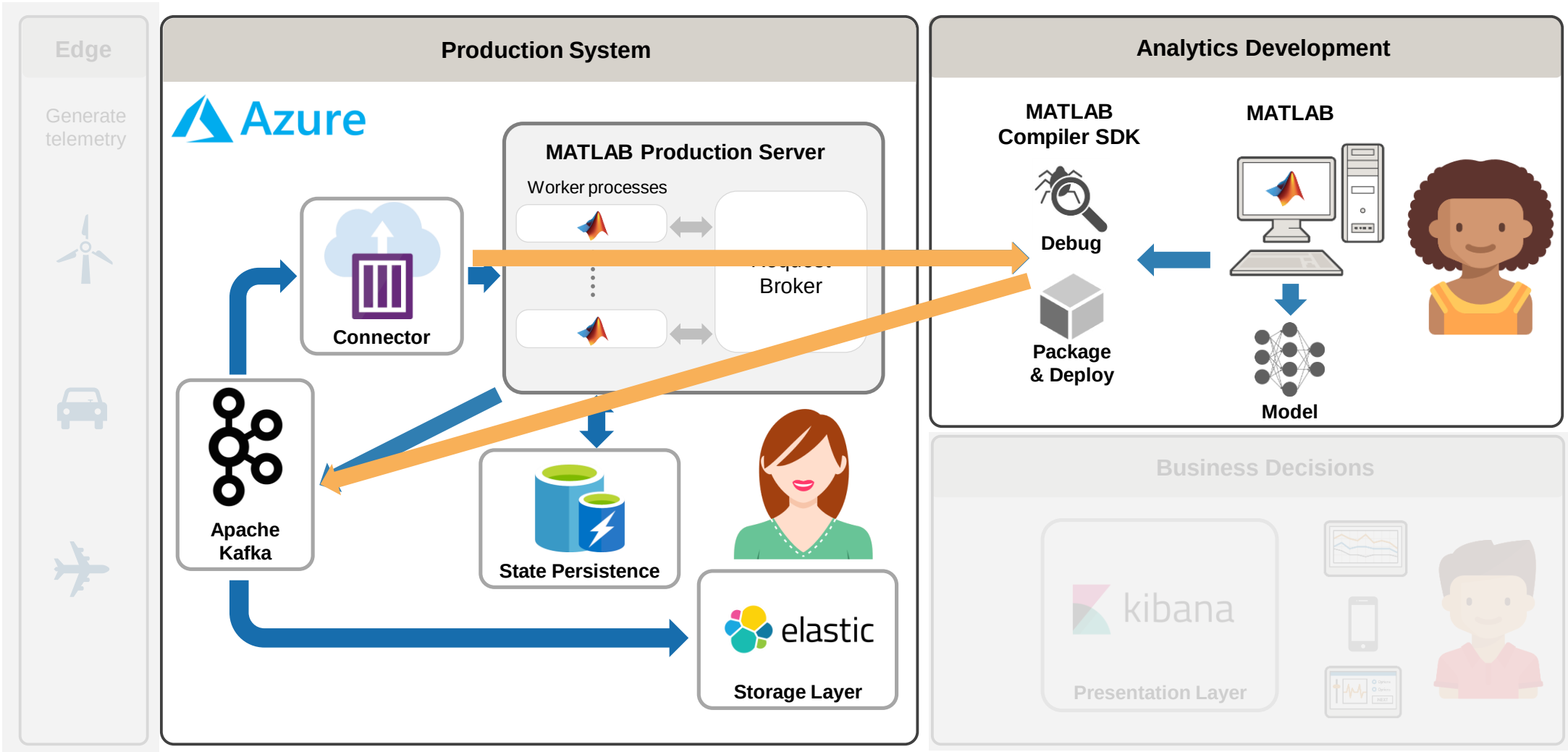
Attaching persistence



System Architect

4 Integrate with Production Systems

Debug your streaming function on live data



System Architect

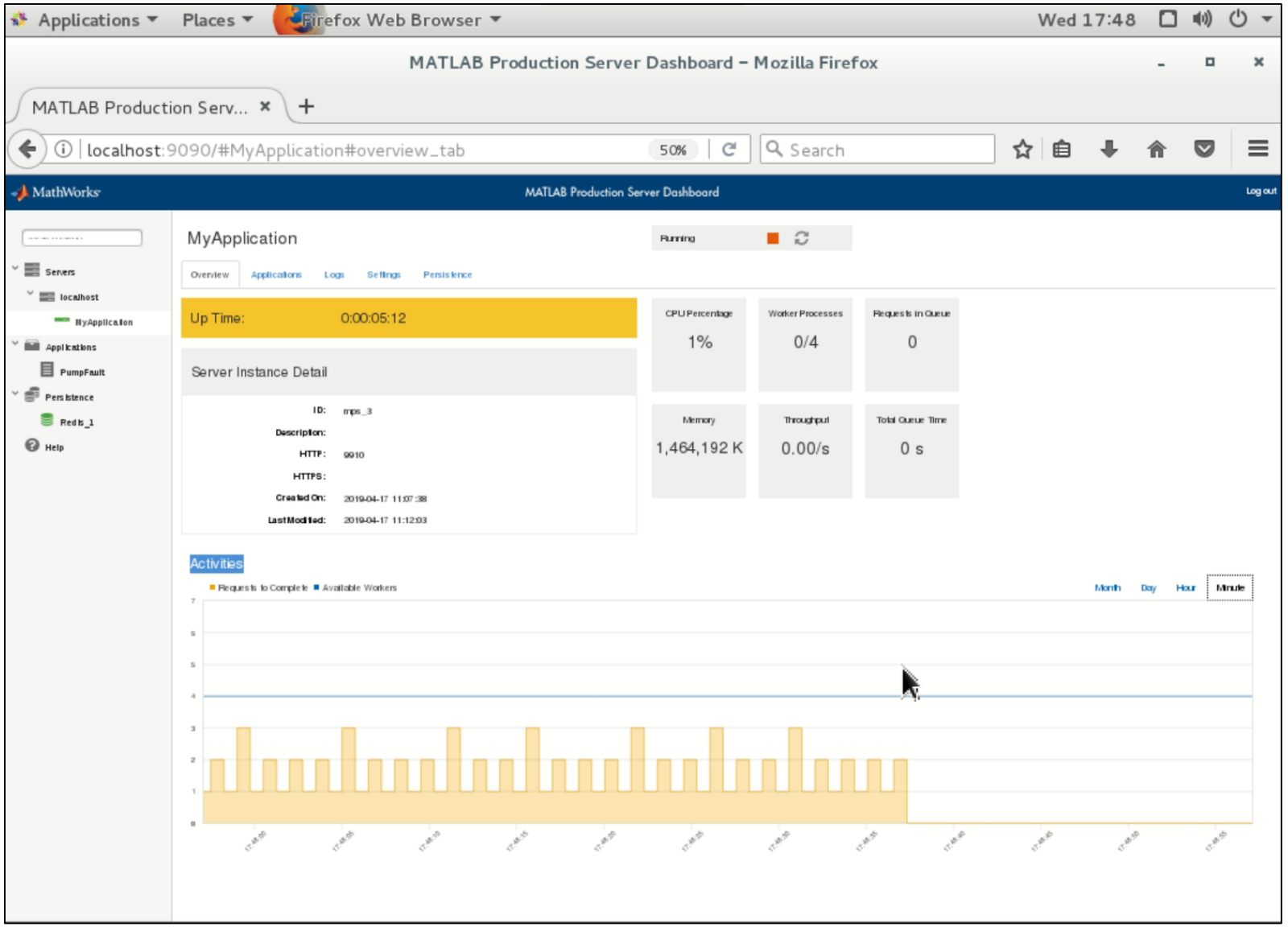
4 Integrate with Production Systems

Debug a Stream Processing Function in MATLAB

System Architect

4 Integrate with Production Systems

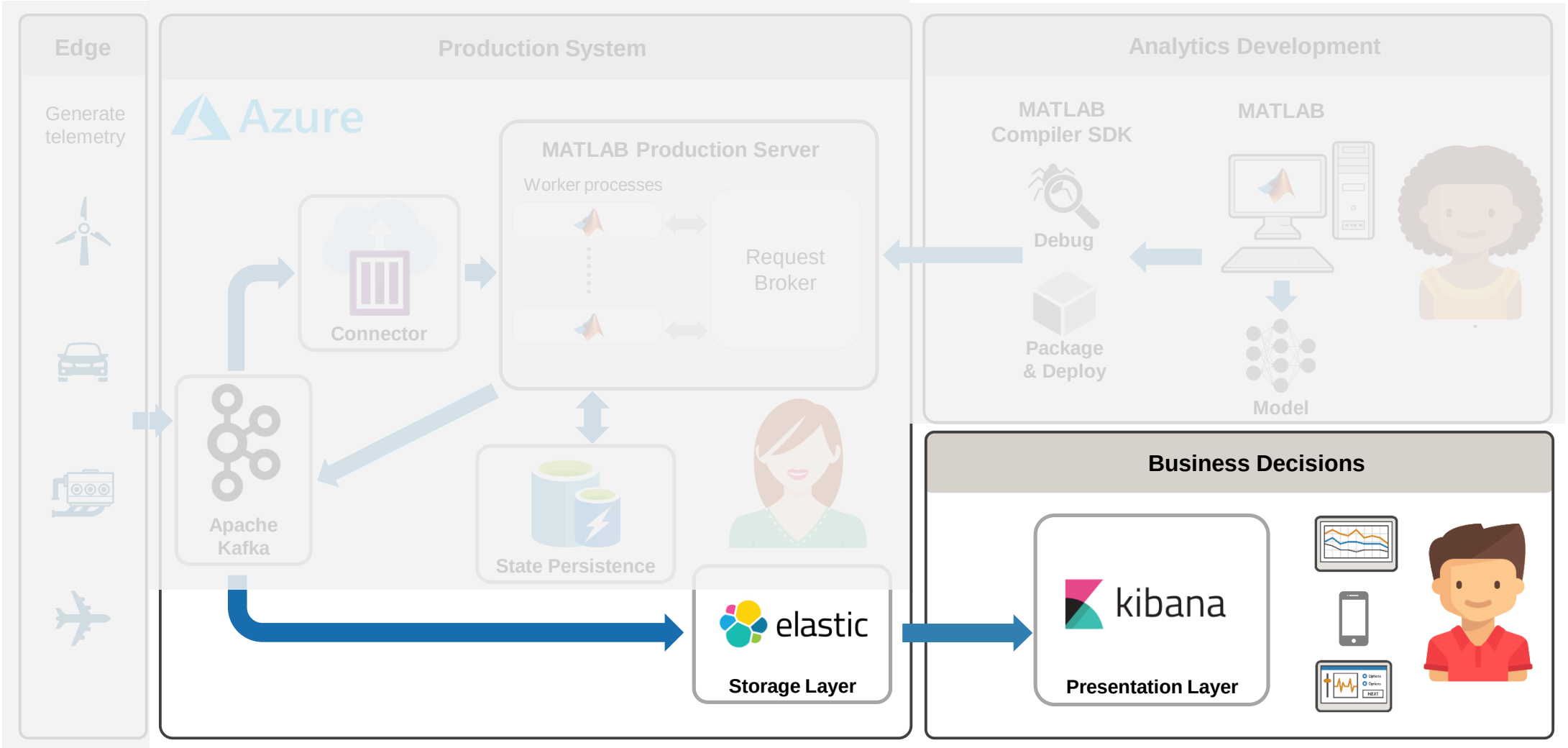
Running Kafka with MPS



System Architect

4 Integrate with Production Systems

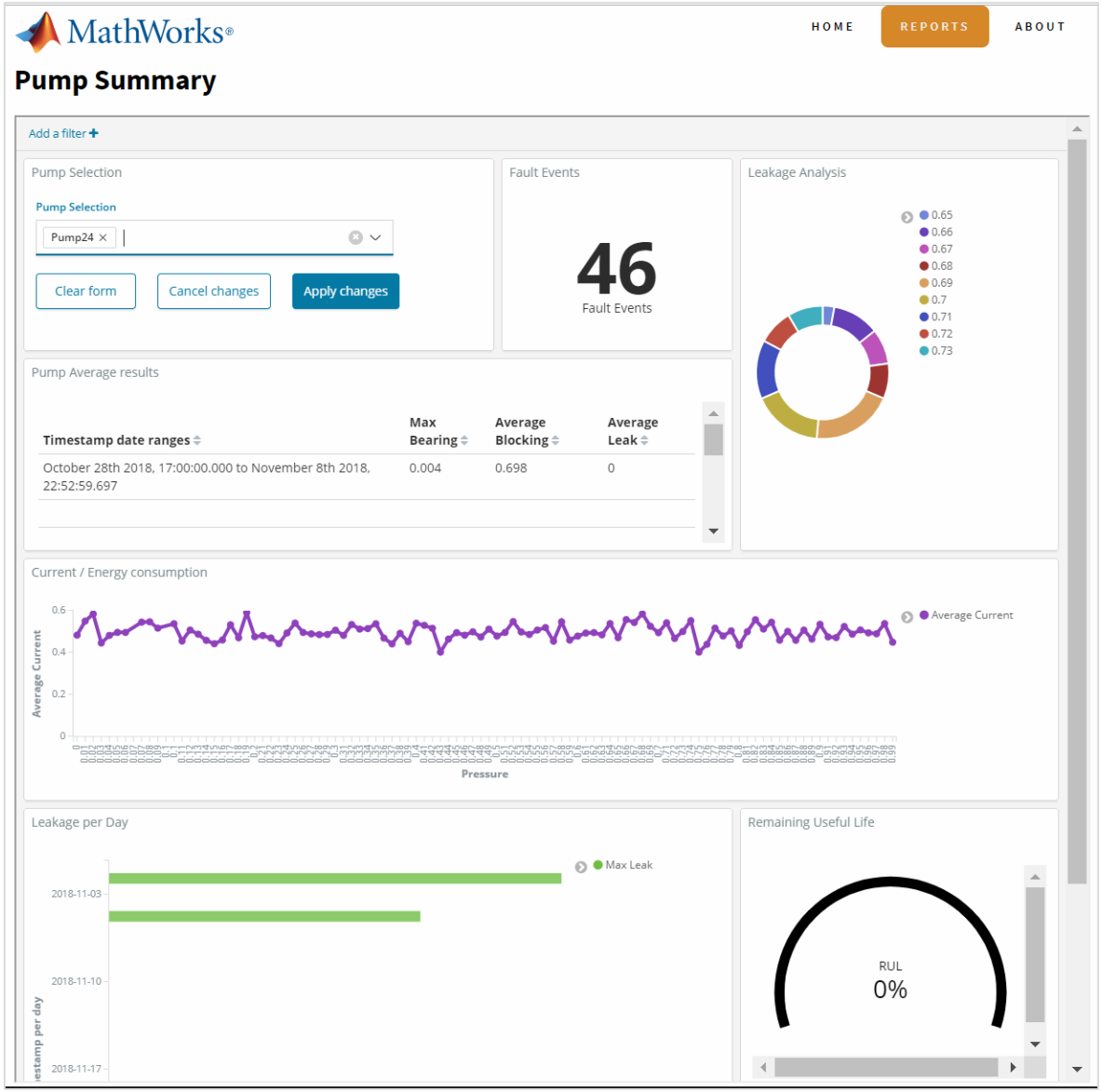
Complete your application



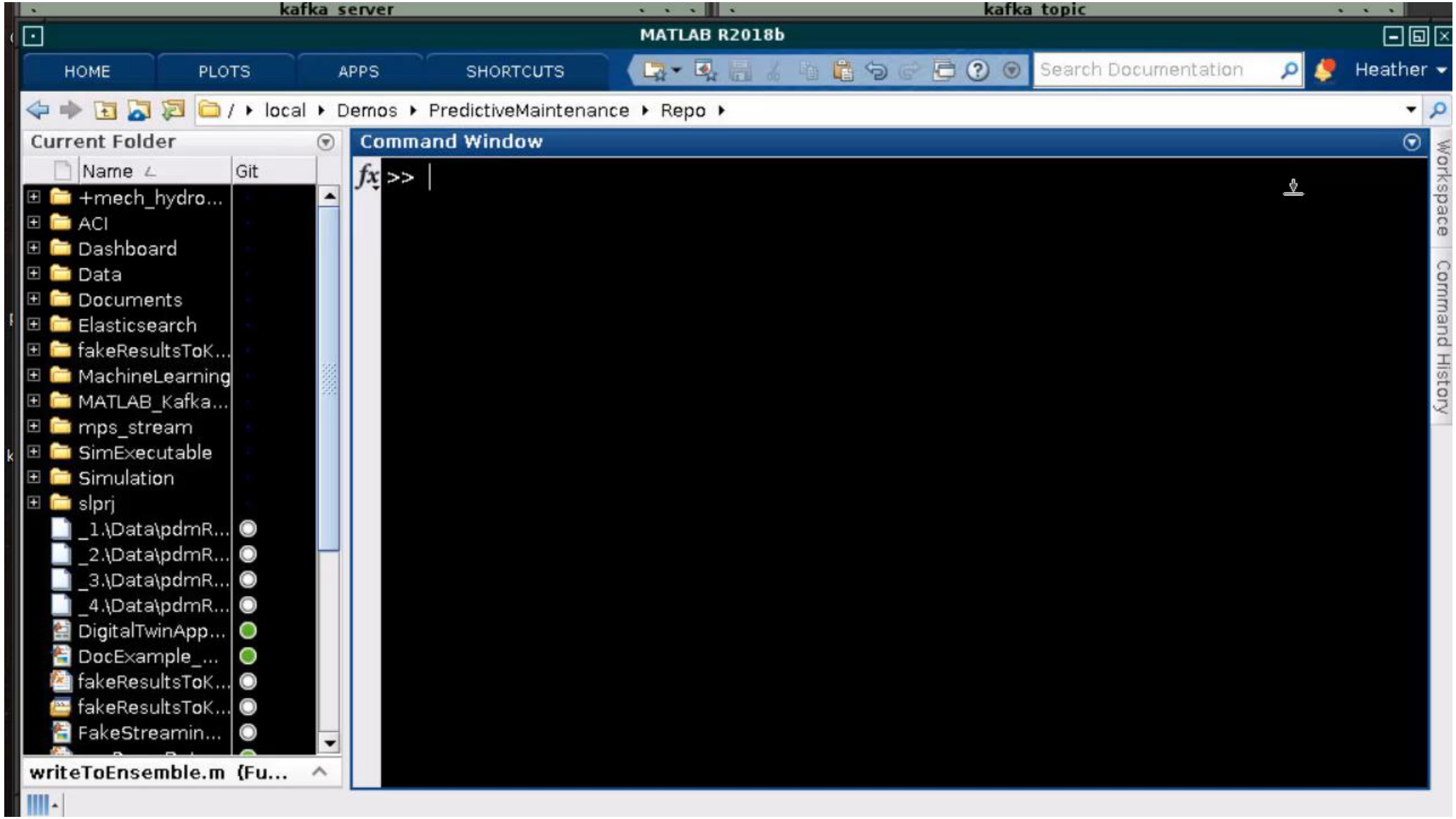
Plant Operator

5 **Visualize Results**

Complete Your Application



Build Standalone UI based applications in MATLAB



MATLAB, Simulink and Cloud Reference Architectures provide

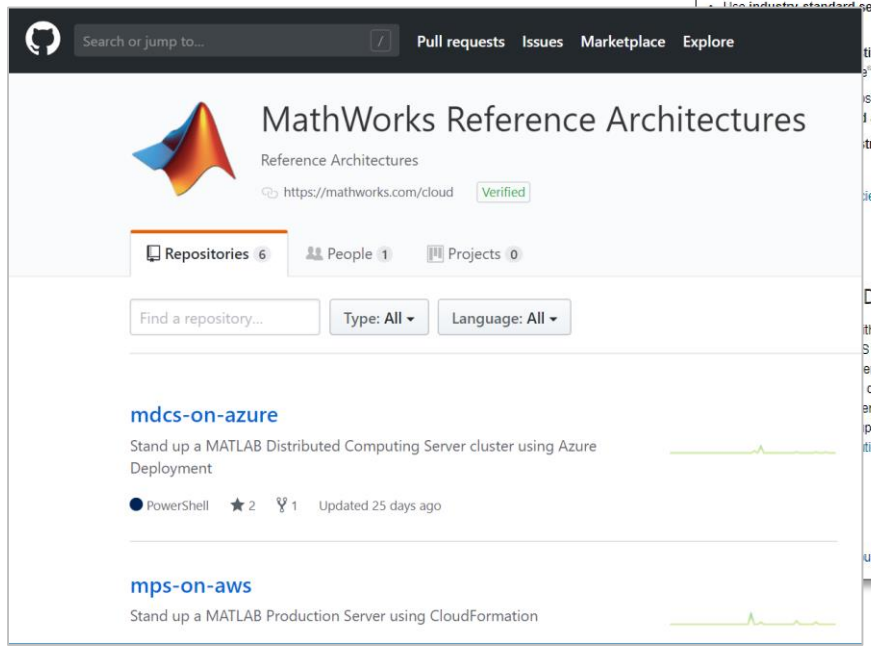
“Integrated AI Development and Deployment Workflow”

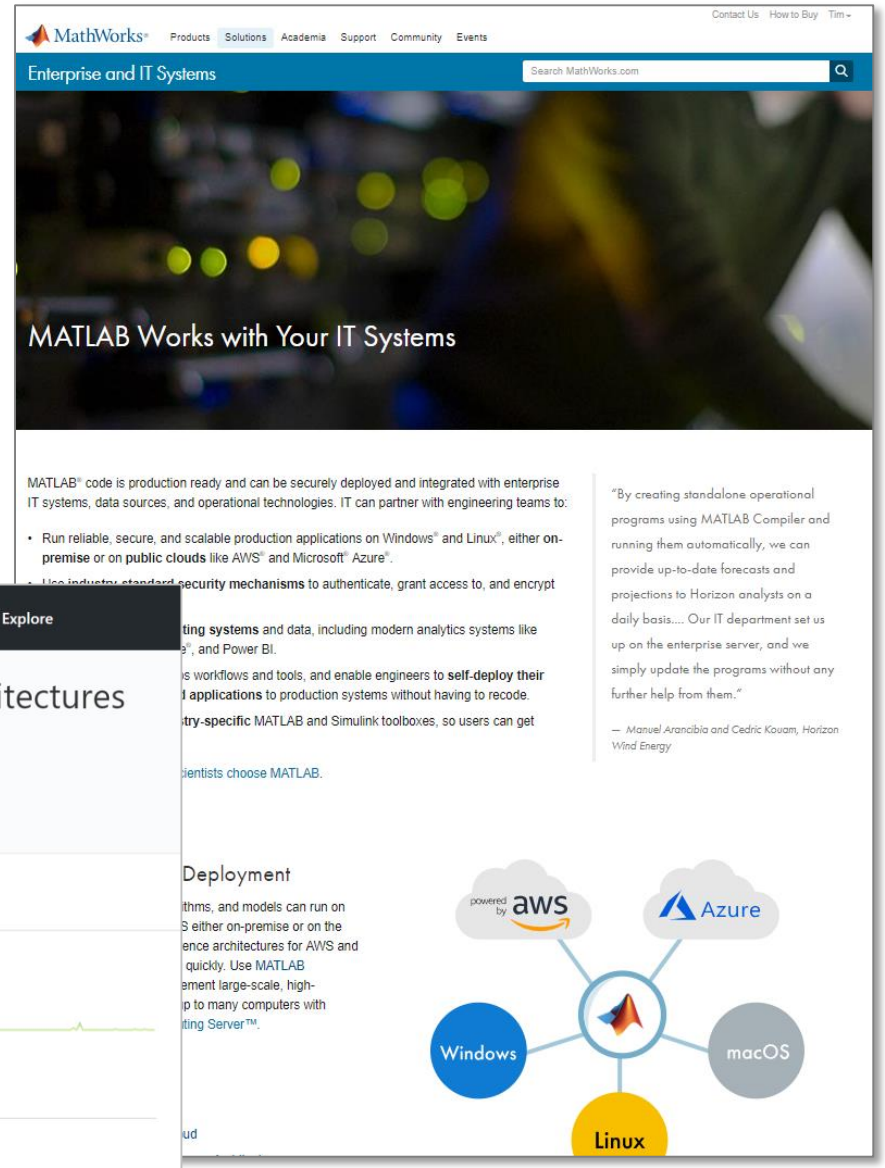
for Cross Functional Teams

- Successfully use Digital Twins to generate faults and train models
- Fast prototyping of physical and AI models with MATLAB
- Easy integration with OSS
- Cloud reference architectures for enabling faster IT setup
- Customize dashboard for Operator’s needs

Resources to learn and get started

- [GitHub: MathWorks Reference Architectures]
- [Working with Enterprise IT Systems]
- [Data Analytics with MATLAB]
- [Simulink]





MATLAB EXPO 2019

Thank You